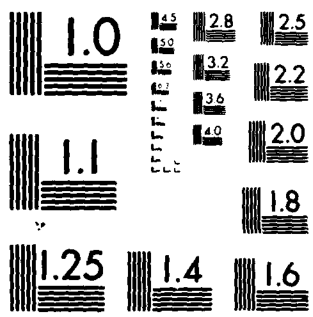
AD-A092 196 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 15/7
THE ENRICHMENT OF SMOLER'S MODEL OF LAND COMBAT.(U)
SEP 80 G M MILLS
UNCLASSIFIED NL

END
DATE
FILMED
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

THE ENRICHMENT OF SMOLER'S
MODEL OF LAND COMBAT

by

Glenn M. Mills

Sep[redacted]80

Thesis Advisor: J. G. Taylor

Approved for public release; distribution unlimited

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A092 196	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) The Enrichment of Smoler's Model of Land Combat		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis; September 1980
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Glenn M. Mills		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE September 1980
		13. NUMBER OF PAGES 82
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Lanchester-type ground combat model.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This thesis provides the student of combat modelling with a computer program for a relatively simple combat model that can be used in a classroom environment for study and analysis. The model is an aggregated, force-on-force ground-combat model that uses

Lanchester's aimed-fire equations for casualty assessment. The original version of the model was developed in 1979 in a previous thesis, and shortfalls in the original model have been overcome, along with the addition of several enrichments to provide added user flexibility. A user's manual is provided to facilitate user access to the model from a permanent disk in the W. R. Church Computer Center.

Accession For	
NTIS GRA&I	X
DDC TAB	
Unannounced	
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or special
A	

Approved for public release; distribution unlimited

The Enrichment of Smoler's Model of Land Combat

by

Glenn M. Mills
Captain, United States Army
B.S., North Carolina State University, 1971

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1980

Author Glenn M. Mills

Approved by James G. Taylor
Thesis Advisor

Glenn F. Lindsay
Second Reader

Chairman, Department of Operations Research

W M Woods
Dean of Information and Policy Sciences

ABSTRACT

This thesis provides the student of combat modelling with a computer program for a relatively simple combat model that can be used in a classroom environment for study and analysis. The model is an aggregated, force-on-force ground-combat model that uses Lanchester's aimed-fire equations for casualty assessment. The original version of the model was developed in 1979 in a previous thesis, and shortfalls in the original model have been overcome, along with the addition of several enrichments to provide added user flexibility. A user's manual is provided to facilitate user access to the model from a permanent disk in the W. R. Church Computer Center.

TABLE OF CONTENTS

I. INTRODUCTION

As the art of combat modelling becomes more advanced, combat modelers are continually building more and more complicated models. To the beginning modeler, the ability to understand how these models operate is difficult, if not impossible. As a student of combat modelling, I have sensed a need for a relatively simple model that could be easily studied in a classroom environment. It was with this in mind thst this project was initiated.

At present, there seems to be no simple combat model available that demonstrates the basics of model building to the beginning student. In 1979, J. Smoler (a student at the Naval Postgraduate School) attempted to build such a model for his thesis research (Smoler,1979). His model was a deterministic, force-on-force computer model that used Lanchester's aimed-fire equations for casualty assessment. The general scheme of his model is shown in Figure 1.

Although Smoler's model was a bold attempt at a simplistic combat model, it did have some problem areas that warranted investigation. Some of these problems were first discovered during a class project in a combat models class (OA 4655) at the Naval Postgraduate School. It is the purpose of this thesis to study Smoler's model and to undertake an enrichment program that will make the model

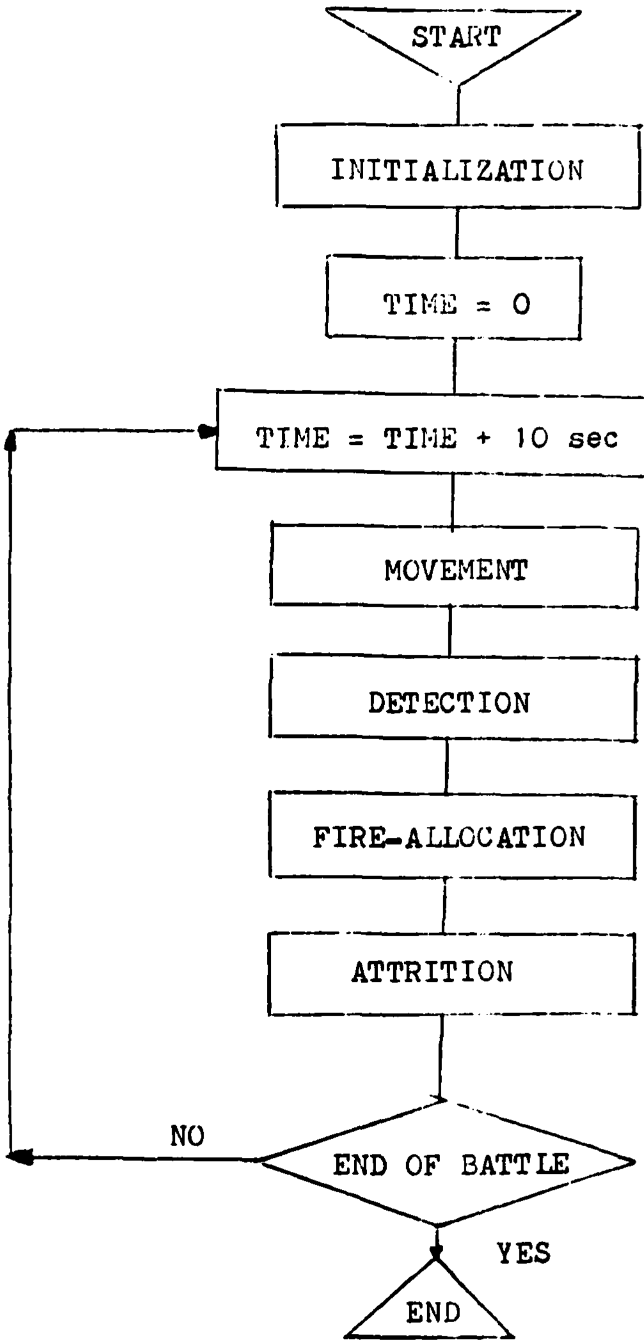


FIGURE 1. GENERAL SCHEME

more realistic and flexible, while maintaining it's transparency and simplicity.

Once enhanced, the model would then be made available for use as an instructional tool for combat modelling classes. This will be accomplished through the development of a user's manual to facilitate the use of the model by students by placing the model on a permanent disk in the W. R. Church Computer Center where it will be easily accessed by any desired user.

The remainder of this paper will discuss, in detail, the problems found with Smoler's model and the methods used to solve these problems. It will also outline some new features that have been incorporated into the model to allow user flexibility. Finally, the User's Manual (Appendix A) will provide all the required information to enable even the novice user to utilize the model.

II. THE ORIGINAL MODEL

A. GENERAL DESCRIPTION

Smoler's original model of land combat is a deterministic model that plays combat between two homogeneous forces, a blue force and a red force. The blue force is comprised of three subunits in a static defense, with each subunit armed with three TOW antitank missile systems. The red force is composed of three subunits of three tanks each, attacking on pre-planned routes. The battle takes place on the 10 x 10 Km Fulda Box that has been developed and used in the STAR simulation model (Wallace, 1978). Since the major thrust of this paper is to alter the original model to a form that will be easily used for classroom instruction, a brief discussion of the major components of the original model will be presented, including problem areas that have motivated model changes.

B. ATTRITION PROCESS

The attrition process in the original model utilizes Lanchester "aimed-fire" equations with variable attrition coefficients. The Lanchester equations used are fairly simple and will be discussed later. However, the calculation of the attrition coefficients is of more immediate interest. Smoler used the Bonder-Farrell model to calculate the coefficients, A_{ij}, the rate at which one firer

of unit i kills unit j targets. These A_{ij}'s are computed according to:

$$A_{ij} = 1/E(T_{ij})$$

where $E(T_{ij})$ is the expected time for one firer of unit i to kill one target of unit j. The $E(T_{ij})$ is computed using the Bonder-Farrell formula:

$$E(T_{ij}) = t_a + t_1 - t_h + (t_h + t_f)/P(K|H) + (t_m + t_f)/P(h|m) \times ((1-P(h|h))/P(K|H) + P(h|h) - P)$$

where

t_a = time to acquire a target

t_1 = time to fire first round following acquisition

t_h = time to fire following a hit

t_m = time to fire following a miss

t_f = time of flight of a round

P = probability of a first round hit

$P(h|h)$ = probability of a hit following a hit

$P(h|m)$ = probability of a hit following a miss

$P(K|H)$ = probability of a kill given a hit

This formula holds for the conditions that the hit probability of any round depends only on the result of the previous round and no accumulated damage is considered. Smoler also assumed that for the TOW weapon system, $P(K|H) = 1.0$ and $P(h|m) = P(h|h) = P$, thus reducing the

equation to:

$$E(T_{ij}) = t_a + t_l + t_f + (t_m + t_f)(1-P)/P$$

Smoler also assumed that for tanks, $P(K|H) = 1.0$ and $t_f = 0$, reducing the formula to:

$$E(T_{ij}) = t_a + t_l + t_m(1-P)/P(h|m)$$

Utilizing these equations to calculate the A_{ij}'s, the attrition during each time step was computed using the Euler-Cauchy differencing equations to approximate Lanchester's force-on-force attrition differential equations.

C. UNIT LOCATION AND MOVEMENT

Red and blue unit locations were handled in two different ways in the original model. The blue locations were left as user inputs, while the red locations had been pre-determined by the model builder and could not be altered by the user. This allowed flexibility of defensive positions, but because of the method of determining movement routes for the attackers, this flexibility was limited.

The method utilized for route determination was, at best, unrealistic. For each original red location, a straight west to east route was calculated. Each route was divided into 40 meter intervals, since each red unit was assumed to move that distance during each 10 second time interval. This approximated an average rate of movement of 9 m.p.h. This method of route selection, although easy to implement, has several significant shortcomings. It does not allow the attacking units to utilize terrain

features during movement. Also, it permits attackers to move over terrain that is, in real life, impassible.

D. BATTLE TERMINATION

In any combat model, adequate battle termination rules must be considered. Smoler utilized two criteria to terminate the battle. The first of these was annihilation of one of the two forces. This criteria is reasonable in an expected value model like this and was adequately handled. This is not the case for the second termination rule.

The second rule for termination is that the distance between red and blue forces becomes too small. The use of this criteria for a stopping rule is fine, provided it is implemented properly. In the original model, a geographic center was calculated for each force. It was these center of mass points that were compared to determine if the units were too close. This method can lead to problems, such as those that were encountered during some initial trial runs. In one battle, the red units, during their advance, had eliminated one entire blue unit and had suffered the loss of one of their own units. Since the red units are allowed to advance as much as 150 meters apart and blue units were set up so they were approximately 1300 meters apart, there was considerable distance between units on the battlefield. Since the two units had been destroyed, the center of mass computation allowed the red

units to move completely through the blue defense, while both sides still had forces remaining to fight with. This reflects a problem with the center of mass computation as a method for determining distances between units.

E. DETECTION AND FIRE ALLOCATION

Both detection and fire allocation processes are handled well in the original model and no changes have been made. A detailed description of these methods is contained in Smoler's Thesis(Smoler,1979). However, a brief and general discussion of both processes will be included here.

The detection phenomena is modeled in two ways. First, a non-firing detection can occur as a result of an observer's random search within his designated section of responsibility. A 30° field of vision for an observer is assumed, and the probability that an observer is looking in the direction which enables him to detect a target is computed by integrating the Limicon Function over limits that are $\pm 15^{\circ}$ from the primary direction the observer is looking. The Limicon Function, $f(\theta)$, is the following probability density function:

$$f(\theta) = A + B \times \cos(\theta)$$

where

D = assigned sector width/2

$A = -B \times \cos(D)$

$B = 1 / 2(\sin(D) - D \times \cos(D))$

θ = primary direction observer is looking

Also, A and B are chosen such that

$$\int_{-D}^{D} f(\theta)d\theta = 1$$

The second method of detection is a firing detection. This phenomena occurs when the following happens. If a firer's location is within $\pm 15^{\circ}$ of an observer's primary direction of observation when he is firing, he is assumed to be detected and is added to the observer's target list. This models the detection of a firer by locating a firing signature of a weapon.

The fire allocation process is also modeled well by Smoler. Since each firing unit is not restricted to firing at one target, a fraction of each firing unit is allocated to each target on the firer's target list. This fraction is determined as a function of range to each target and of predetermined priorities of fire that are set so each unit's fire priority goes to those targets to his immediate front. Targets to the firer's flanks are then allocated a smaller percent of the available firepower. Again, the detection and fire allocation processes have been handled well and no changes to these routines will be made.

III. MODEL CHANGES

A. GENERAL

As the previous section outlines, there were several major problems discovered in Smoler's original model. In order to make the model more flexible and classroom useable, several major changes were found to be necessary. This chapter will provide a detailed description of these changes, which include conditionally-deterministic attrition, unit locations and movement, battle termination and movement of defenders.

B. CONDITIONALLY-DETERMINISTIC ATTRITION

In order to introduce a stochastic process into the attrition computation for this model, several options were available. These include the use of a Markov-process to determine casualties or the use of random attrition-rate coefficients. After considering both options, and in keeping with the transparent nature of this model, it was decided that stochastic attrition coefficients should be utilized, because the attrition-rate coefficient is a random quantity measuring a unit's fighting ability and can be realized before any given battle. The following procedure was used to develop a method for random attrition-rate coefficients.

The attrition-rate coefficient, A_{ij}, is a measure of the rate a firer of unit i attrits a target of unit j.

This also can be interpreted as a measure of the fighting ability of an element of type i. It is intuitively obvious that this is a variable quantity that can be affected by many different factors, including weather, espirit-de-corps, previous engagements and leadership, to mention only a few. In a homogeneous model, one can attempt to capture these random effects by developing a distribution of initial fighting unit capabilities, i.e., initial A_{ij}'s for each unit. Since no data is readily available that captures this phenomena, an attempt was made to fit a distribution to an intuitive feeling as to how this fighting ability varies from one unit to the next prior to a given engagement. It was the author's intuitive feeling that the A_{ij}'s should be distributed between .3 and .8 with the majority of the units being rated between .5 and .6.

Utilizing the above intuition, an attempt was made to "fit" a distribution to these assumed fighting levels. Graphically, the distribution would look like Figure 2.

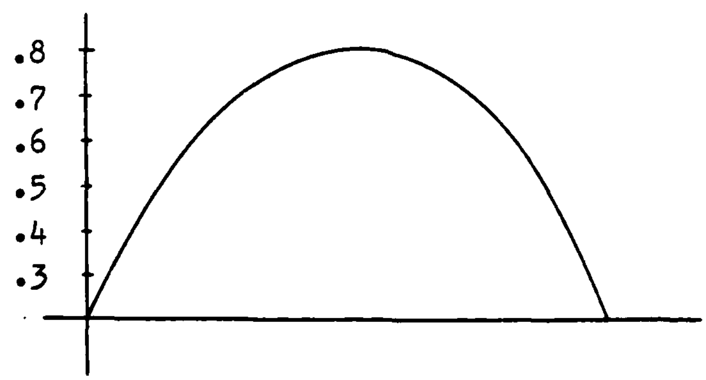


Figure 2. Distribution of Initial A_{ij}'s

Initially, an attempt was made to fit a Beta distribution to this curve. The fit was fairly accurate in the middle range, but was not satisfactory in the tails. The same was true for a Normal approximation. It was finally determined that the best fit was a straight quadratic fit, where a Uniform(0,1) input variable could be used and the output would yield the desired random number. The quadratic fit that was used is:

$$A^{o}_{ij} = -2U^2 + 2U + .3$$

Once a distribution for the attrition-rate coefficients had been derived, the implementation into the model was accomplished. Since it was assumed that the fighting ability of each unit is a random quantity prior to a given battle, it was only necessary to obtain a realization of the random variable for each unit prior to the initiation of the battle. This realization, A^{o}_{ij}, is determined by using a random Uniform(0,1) number and the above formula. A new attrition-rate coefficient for each unit is then computed during each time step using the equation:

$$A_{ij} = \begin{cases} A^{o}_{ij}\ (1-r/r_e)^2 & \text{for } 0 \le r \le r_e \\ 0 & \text{for } r_e \le r \end{cases}$$

where

r_e = maximum effective range of a firer's weapon

r = current range between firer and target

A^{o}_{ij} = realization of unit's fighting ability

This equation is utilized because it varies with range, but it also is a function of A_{ij}^{o}, thus creating a different attrition-rate curve for each unit, depending on that unit's fighting ability prior to the battle. Graphically, as an example, the attrition-rate coefficient curve for an A_{ij}^{o} of .6 and a maximum effective range, r_e, of 3000 meters would look as in Figure 3.

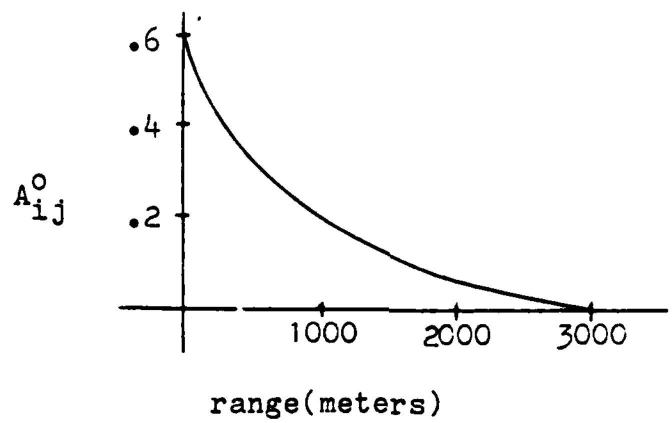


Figure 3. Attrition-rate coefficient curve for $A_{ij}^{o} = .6$ and $r_e = 3000$ m.

Once the above method for determining attrition-rate coefficients had been selected, it was coded and included as a user option in the program. Once the code had been implemented, several runs were made utilizing different random number seeds to compare battle results using this method for attrition-rate coefficient determination. These runs were then compared with a run using the original deterministic method. The results are summarized below.

First, the model was run using the deterministic attrition method. For purposes of comparison, all locations,

movement rates, force levels and other input variables were held constant for all runs. Combatant attrition using the deterministic process as a function of time is shown in Figure 4.

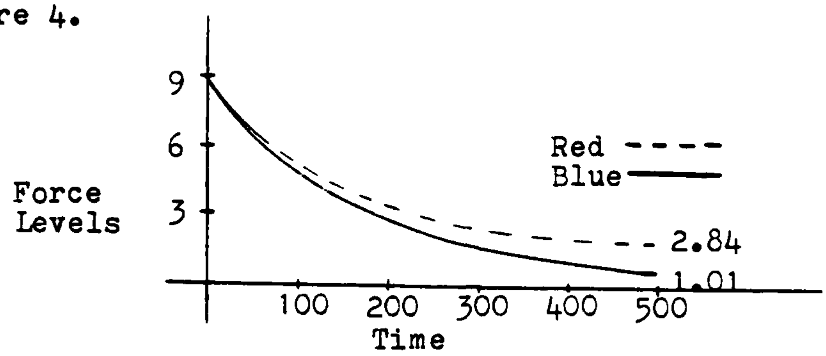


Figure 4. Attrition using deterministic attrition

In this battle, termination occurred as a result of forces being too close together, with force levels at termination reflected in Figure 4. This was then compared to the runs that utilized the new attrition module. The results of four of these runs are in Figure 5. The A^o_{ij}'s are the realizations of the random variable denoting a unit's initial fighting capability prior to the battle. The three numbers under the red and blue headings are the realizations for each subunit in the battle on which casuality assessments are computed.

These results show, in fact, that a randomly selected A^o_{ij} does have an effect on the final outcome of the battle. This module now provides the user the option of selecting a conditionally-deterministic process that still uses Lanchester's equations as the basis for the attrition computation.

A^o_{ij}		Force Level			
Blue	Red	Blue	Red	Time	Result
.33 .77 .46	.79 .49 .56	1.7	1.5	780	Too close
.39 .67 .58	.63 .78 .36	4.3	0	330	Blue wins
.78 .67 .80	.74 .68 .72	3.2	0	320	Blue wins
.32 .43 .68	.51 .77 .65	1.2	3.8	640	Too close
Deterministic		1.1	2.8	460	Too close

Figure 5. Comparison of Stochastic and Deterministic Battles

C. UNIT LOCATION AND MOVEMENT

As was pointed out in an earlier section, the unit locations and movement logic that Smoler used tended to be rigid and inflexible. The changes that have been made to rectify this and allow more flexibility for the user will be discussed here.

First, the user is now responsible for inputing all combatant locations, both attackers and defenders. The format for these inputs will be specified in Appendix A, the User's Manual. Aditionally, the user is given the option of selecting routes for the red advance or allowing the model to calculate straight west to east routes as in the past. To add more flexibility, the user will also be required to input the rate of advance (vehicle speed) for the attacking forces. The inputed rate of advance is used in both methods of route calculation, thus controlling the red attack.

1. Straight Line Routes

The option to use straight line routes from west to east is handled in the same manner as before. The route is broken down into discrete distance intervals from the initial red location straight to the east. The only difference is that the length of each interval is determined by the vehicle speed that the user inputs. The route for each red unit consists of 125 equal length intervals.

2. User Determined Routes

The option to allow the user to select routes for the red advance has been added to enable the user to make use of available terrain and to add realism to the model. The method for calculating these routes is straightforward and is discussed next.

The user is required to input the original location of each red subunit and from one to ten nodes he wishes each attacking subunit to move through. This information, along with vehicle speed, is used to calculate route intervals that move the attacking unit through each of the designated nodes. A complete route would look like the one depicted in Figure 6. The method used to compute the routes is as follows.

The straight line ground distance between the first two adjacent nodes, DIST, is calculated as shown in Figure 7. The angle between the desired direction of movement and straight west to east movement, a, is then calculated. Utilizing these quantities and the distance desired to

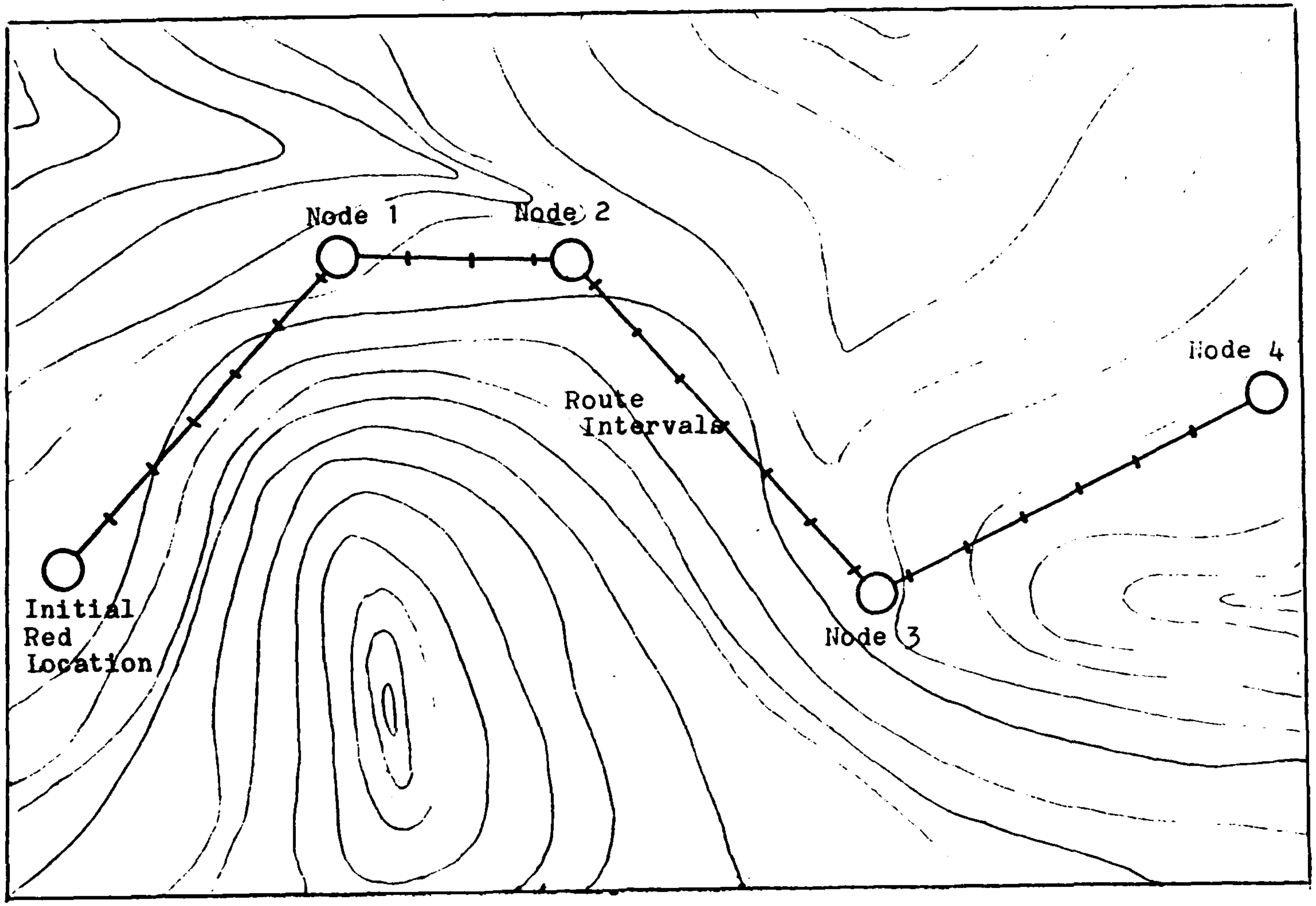


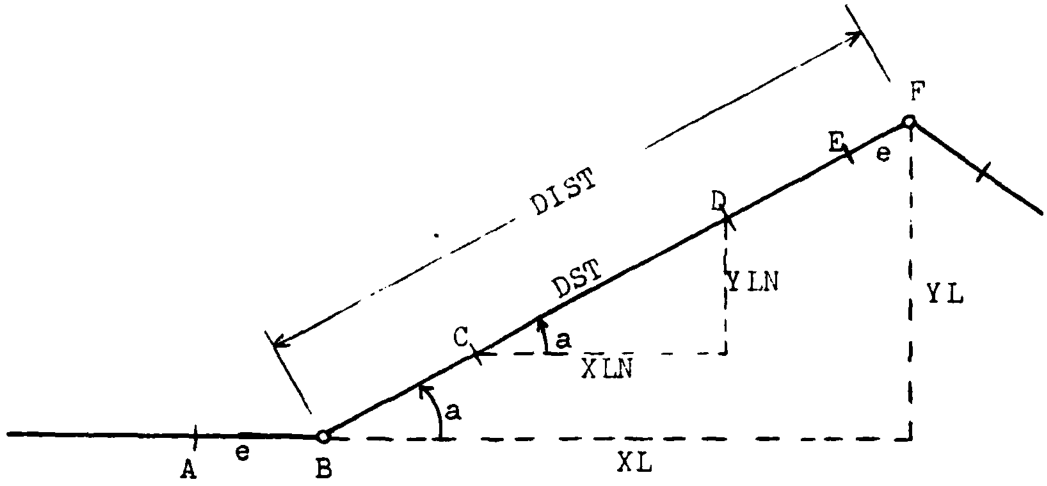
Figure 6. User Determined Routes

move during each time step, DST, the distance to be moved in the x and y direction, XLN and YLN, is now computed as shown in Figure 7. These distances are then added to the coordinates of the previous interval endpoint, point C in the figure, to determine the coordinates of the next interval endpoint, point D. This same distance is again added to compute the coordinates of the next endpoint, point E. This process is continued until the distance from the last endpoint computed to the next node is less than DST. This process is repeated between the next two nodes until the entire route is completed. In order to insure all intervals are of equal length, the computation of the first interval between any two nodes must be considered separately, by taking into account the distance left over from the last computation between the previous two nodes. To do this, the first interval takes the remaining distance, e, and adds it to an interval length of DST-e for the first interval between any two nodes. This insures that each interval along the route is of length DST, which is the required length.

D. BATTLE TERMINATION

As previously stated, the battle is terminated either by annihilation of one of the two forces or by forces getting too close. The annihilation criteria has not been changed. However, to insure the distance criteria is effective, two changes have been made.

First, the distance between units is no longer calculated



NODE	COORDINATES	USE
A	XIC(N-1),YIC(N-1)	$(N-1)^{st}$ Interval Endpoint
B	XLOC(I,J)	User Inputed Node
C	XIC(N),YIC(N)	N^{th} Interval Endpoint
D	XIC(N+1),YIC(N+1)	$(N+1)^{st}$ Interval Endpoint
E	XIC(N+2),YIC(N+2)	$(N+2)^{nd}$ Interval Endpoint
F	XLOC(I,J+1)	User Inputed Node

$DIST = \sqrt{XL^2 + YL^2}$

$a = \tan^{-1} (Y/XL)$, where $Y = |YL|$

e = distance less than DST at end of calculation of intervals between adjacent nodes.

$YLN = DST \times SIN(a)$ $XLN = DST \times COS(a)$

Figure 7. Route Computation

as a center of mass distance. Instead, a distance is computed between each attacking subunit on which casualties are being computed that is still alive and each defending subunit that is still in the battle. If any of these distances becomes too small, the battle is considered to have moved to a "close-in" combat (hand-to-hand) that this program does not currently model. For this reason, the battle is then terminated. However, to insure the attackers do not pass by the flanks of remaining defenders and remain outside termination distance, a check of x coordinates for each subunit is also made. If any attacking subunit's x coordinate places him past the location of the forward most blue defensive subunit still in the battle, the battle is also terminated.

The criteria for being too close has been left as a user determination. It is one of the inputs that is described in the User's Manual, Appendix A. This allows for flexibility of breakpoint distances which also lends itself to the study of optimum breaking distances for various weapon systems on the battlefield.

E. ALTERNATE DEFENSIVE POSITIONS

The last addition to the model, added to increase the flexibility and realism of the battle, is the option of having the defending units move to alternate positions if the attackers close to within a user specified range. The move is handled in a simple and transparent manner.

If the user selects this option, the defender will move

when the attacking forces close within the breakpoint distance. When this occurs, each defensive unit that is still alive will move to an alternate position the user has selected. The duration of the move is also a user input. He simply specifies the number of 10 second time steps he wants to allow the blue forces to move. At the completion of the move, during which the red forces continue to attack, the battle will continue until one of the termination criteria discussed above has been met.

IV. FUTURE ENHANCEMENT

Although the above changes have been implemented and a more useable and flexible model has been created, there are several areas that could still be considered in future work. These include the introduction of heterogeneous forces, explicit computation of ammunition expenditure and artillery (indirect fire). Possible methods for employing these ideas are discussed below.

A. HETEROGENEOUS FORCES

The current model involves combat between homogeneous forces only. In other words, each opposing force is comprised entirely of one weapon system type. Added flexibility could be attained by allowing heterogeneous force structures on both sides. This would enable the user to investigate the effect of different force mixes on battle outcome.

The introduction of different weapon systems within a single unit would require extensive changes to the attrition process. Although Lanchester equations for aimed-fire could still be utilized, casualty assesment against each weapon system type by each opposing weapon system type would have to be calculated. The total attrition of any particular unit would then be the sum of the damage assessed to each weapon system of that unit. Aditionally, the force level for each weapon system would be required

as an input as well as a separate set of hit and kill probabilities for each weapon system type against each type of target.

Introduction of this feature would create a more complicated model, adding realism but detracting from the current simple and transparent form. Since the purpose of the current effort has been to maintain this simplicity, this option has not been included, but could be considered for future work.

B. AMMUNITION EXPENDITURE

The combat process is a complicated and intricate process. One of the hardest, and not many times attempted, areas to model is the logistic area. However, in the model currently being studied, the problem of ammunition expenditure could be modeled.

Since the model is an expected value type model, it would not be effective to model the amount of ammunition expended by simply counting bullets as they are fired. Actually, in this type of model it would not be possible to count each round, because of the aggregated nature of each unit. The only way to model ammunition useage is to model it in the same way as attrition. This simply means that each unit would have to be given a starting level of ammunition on hand, a basic load, and each time the unit fired, the expected ammount of ammunition expended would be subtracted from what is on hand. The ammount expended would necessarily be a function of the size of the unit

firing, the number of targets fired at and the rate this unit is firing. This can be directly correlated to the attrition-rate coefficient for the firing unit. When ammunition on hand reaches the zero level, the unit would have to be removed from the battle permanently or for some specified time period, to simulate resupply.

C. ARTILLERY

In any armed conflict, there are more ways to inflict casualties than just direct fire. Many types of indirect fire are utilized on the battlefield, including artillery, close air support and naval gunfire. To include this type of play in this model, primarily artillery, would involve a major effort with large program additions.

To model artillery in an expected value model that already uses Lanchester equations for direct fire would suggest the use of Lanchester's area-fire equations. These equations are no more complicated to handle than the aimed-fire equations that are already in use. However, other model considerations would have to be investigated. These would include location of artillery units, whether to use forward observers for target location or only use pre-planned strikes or both and fire allocation procedures to be used. Also, a significant ammount of data would need to be collected concerning weapon types to be used, effective ranges, killing radii and accuracy data.

To model artillery would be a significant addition to the model. However, this too would effect the simplicity and transparency the model currently possesses. This addition has also been left for future consideration.

APPENDIX A

User's Manual

I. INTRODUCTION

Smoler's model of land combat is a force-on-force combat model that utilizes Lanchester's aimed fire equations for casualty assessment. The battle is simulated on a 10 x 10 KM piece of terrain representing an area east of Fulda, West Germany. A portion of the terrain map is inclosed as Figure 1. It is a computer model that is coded in FORTRAN and is available for use in the W. R. Church Computer Center.

The purpose of this manual is to familiarize the user with the model and to provide the required Job Control Language (JCL) and user inputs to run the model. A sample listing of program output will also be provided to familiarize the user with the expected program output.

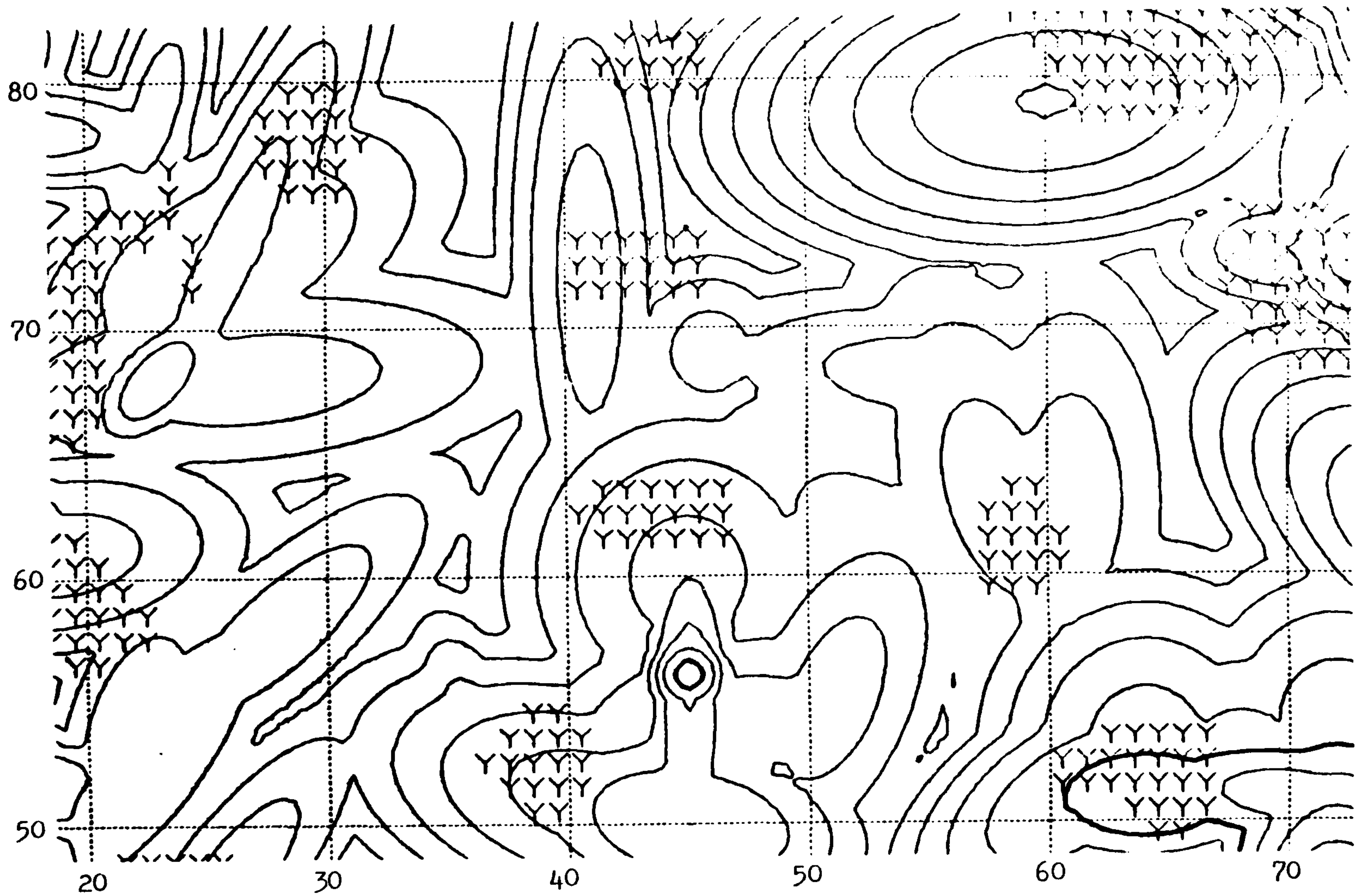


Figure 1. Model Terrain Map

II. PROGRAM STRUCTURE

Smoler's enhanced model is a computerized model, coded in FORTRAN, containing a main program and nine subroutines. To assist the user in understanding how the model operates, a brief description of the function of each of these major parts of the program will be included.

A. MAIN PROGRAM

The main program has several important functions. All of the input and output functions, except the line-of-sight data, are contained in the main program. Aditionally, the main program is used to structure all of the other functions during each 10 second time step. Attrition, detection, movement and fire allocation are also handled in the main program. The nine subroutines provide needed input numbers for the above calculations. The general flow of the program is depicted in the flow chart shown at Figure 2.

B. SUBROUTINES

There are nine subroutines included in the model. Each of these subroutines performs a distinct function, each of which will be discussed below.

1. Subroutine SETUP

This subroutine is used to read in the terrain data for the Fulda Map. This terrain data will be used when computing line-of-sight between targets and observers,

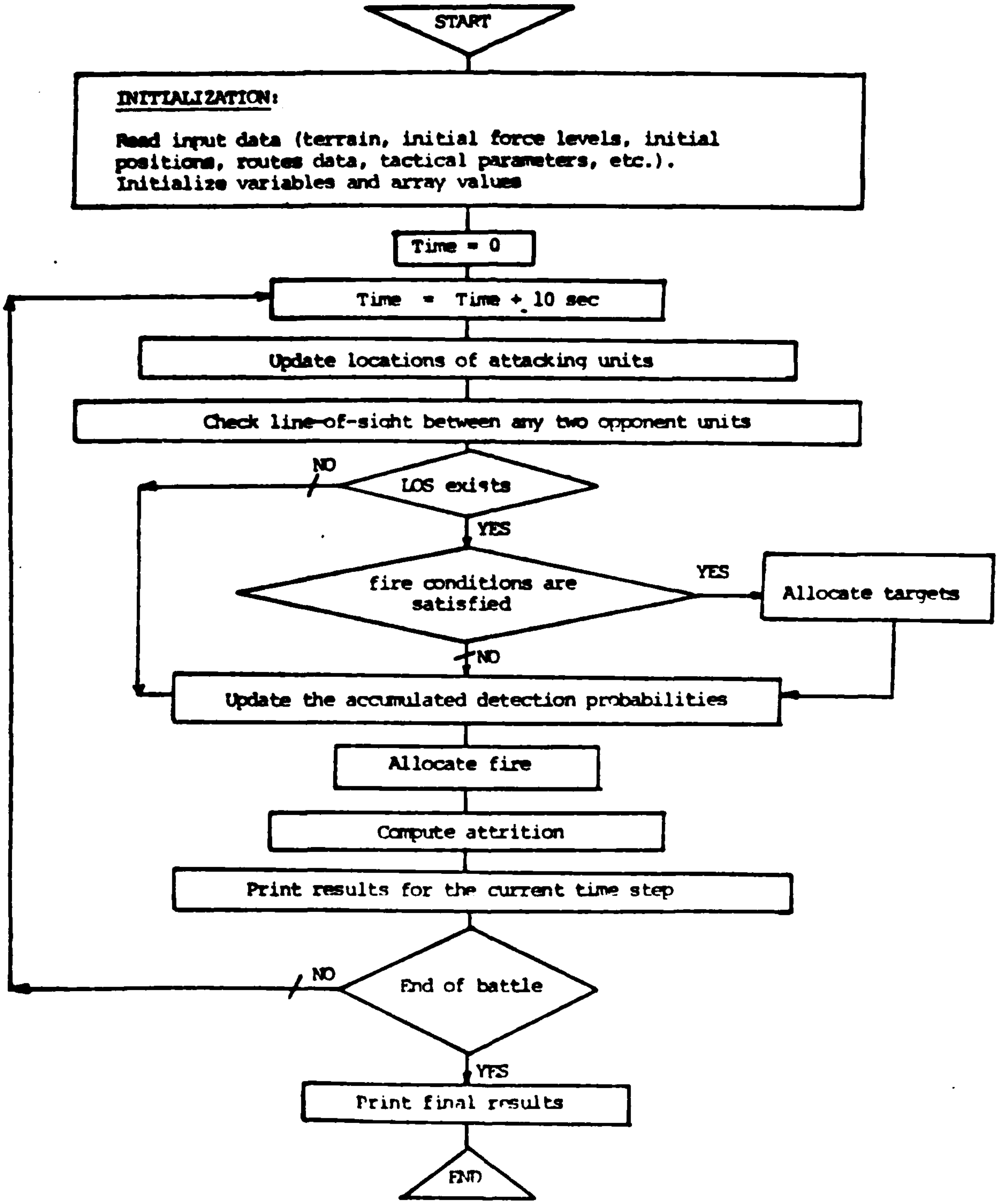


FIGURE 2. FLOW CHART

as well as providing a grid system for unit locations and movement.

2. Subroutine LOS

This module, developed by Professor Jim Hartman of the Naval Postgraduate School, computes a percent of a target visible to a particular observer, given the coordinates (location) of both. This visible fraction is used in the detection and attrition modules in the model.

3. Subroutine KCVER

This subroutine is used by subroutine LOS in determining what portion of a particular target is covered by the terrain between the target and observer. This number is used in the detection and attrition modules.

4. Subroutine ETK

This module computes the expected time for a given firer to kill a given target. The calculation is a function of range, time of flight for a round and hit and kill probabilities for the firing weapon system. It is a number that is used in the computation of the deterministic attrition coefficients, A_{ij}.

5. Subroutine STOCH

This is used when a user has selected a stochastic attrition option to compute the attrition coefficients during each time step. The calculation is a function of the original stochastically determined A_{ij} as well as a function of range.

7. <u>Subroutine LAMBDA</u>

This subroutine is used in conjunction with the LOS routine to compute the detection rate of a target by an observer given the percent of target visible to the observer.

8. <u>Subroutine ROUTE</u>

This is used to compute the route of each attacking red unit when the user has selected the option of inputing attacker routes. It calculates the coordinates of each interval endpoint along the route, making each interval length (distance moved during a 10 second time step) the same. The interval length is determined by the speed the user has selected and inputed for the current battle.

9. <u>Subroutine ELEV</u>

A subroutine that is used to calculate the terrain elevation for a given set of X, Y coordinates. This function is used in conjunction with the LOS subroutine in computing line-of-sight between observer and target.

III. AVAILABLE OPTIONS

The enhanced version of this model has been written to allow for maximum user flexibility while maintaining the simple and transparent nature of the model. To allow this flexibility, there have been several user options incorporated into the model. Each of these options, including user responsibilities, will be discussed here, with the required input data for each being outlined in the next section.

A. STOCHASTIC VS DETERMINISTIC ATTRITION

The user is required to specify whether he wants to use stochastic or deterministic attrition calculations. Both methods utilize Lanchester's aimed-fire equations, the difference being the method of calculating the attrition-rate coefficients. The deterministic procedure uses the Bonder-Farrell method of calculating the A_{ij}, while the stochastic method selects an initial random A_{ij} for each unit, and uses this as a function of range to calculate the A_{ij} for each time step.

B. ATTACK ROUTES

The second major option available to the user is the method of route computation for the attacking forces. The user has to decide whether to allow the program to compute straight west to east routes or to input the routes

he desires each attacking unit to follow. To select his own routes, a user must input the number of nodes he wishes to have on each route and the coordinates of each of these nodes. The program will then compute routes through each node. The nodes must be inputed in order from west to east and should not create an angle between the west to east axis and the route direction that exceeds 45°.

C. ALTERNATE DEFENSIVE POSITIONS

The third option the user must consider is the use of alternative defensive locations. This option permits the user, if he desires, the capability to move the defenders to alternate positions if the attackers close within some specified distance. This breakpoint distance is decided and inputed by the user and is also used as the distance for battle termination. The alternative to moving the defenders is to terminate the battle when the breakpoint distance is reached.

D. OTHER INPUTS

There are other inputs that are required by the program that the user must provide. These include force sizes, weapon characteristics, unit locations and hit and kill probabilities. The required formats for all inputs are outlined in the next section.

IV. INPUT DATA

All user input data is read in from cards at the start of the main program, MAIN. A brief description of each data element as well as the required format for inputing this data is discussed next. A sample data deck has also been included as Figure 3, with the referenced data element number appended on the right. This sample data deck reflects the use of all data elements. However, not all the elements described below are required, and those that are needed only if a particular option is selected are noted with an asteric next to the data element number.

Data Element	Description and Format
1.	Selection of attrition option and random number seed. A 1 is entered for deterministic or a 0 for stochastic attrition. This number is followed by a five digit random number seed. Format for this card is I1,IX,I5.
2.	Number of blue units (NBU) and number of red units (NRU). Both of these elements are two digit integers. Card format is I2,1X,I2.
3.	Effective weapon ranges listed in the following order. Red minimum, red

maximum, blue minimum and blue maximum. Card format is 4(F6.1,1X).

4. Red force levels for each unit. Card format is F3.1, 1X. The card will contain the number of entries that equals the number of red units (NRU) in data element 2.

5. Type of route computation desired followed by vehicle speed. Use a 1 for user determined routes or a 0 for program determined routes. A one digit entry will designate desired speed as follows:

1 for 9 mph
2 for 12 mph
3 for 15 mph
4 for 18 mph

Card format for these elements is I2, 1X, I2.

6. X, Y coordinates for each initial red location. One card is needed for each red unit with format F6.1, 1X, F6.1.

7.* If the user has selected to enter his own routes in data element 5, the following is required for each route.

a. The number of nodes in the route (from 01 to 10). This card's format is I2.

b. X, Y coordinates of each node along the route in order from west to east. One card for each node in format F6.1, 1X, F6.1.

8. X, Y coordinates for blue location, force level for each unit, primary search direction for that unit and desired search width. Force level must be between 1.0 and 3.0, search direction between 135° and 225°, and search width from 30° to 120°. One card for each blue unit with format F6.1, 1X, F6.1, 1X, F3.1, 1X, I3, 1X, I3.

9. Specify if blue is to move to alternate defensive positions, breakpoint distance, and number of 10 second time intervals allowed for the move. A 1 for no move or 0 for option to move. Format for this card is I1, 1X, F6.1, 1X, I2.

10.* Alternate blue position X and Y coordinates if move is desired (data element 9). One card for each location with format F6.1, 1X, F6.1

11. Red weapon system hit and kill probabilities. The probabilities are entered for 6 range bands, with one card for each range band. Four probabilities for each range are probability of a 1st round hit,

probability of a hit given a previous round hit, probability of a hit given a previous round miss, and probability of a kill given a hit. The 6 range bands are from 0 to 500, 500 to 1000, 1000 to 1500, 1500 to 2000, 2000 to 2500 and 2500 to 3000. Each of these six cards has the format 4(F4.2, 1X).

12. Same as item 11, except the probabilities for the blue weapon systems are entered. Six cards with format 4(F4.2, 1X).

In the formats listed above, a 1X is a space, an Ix is an integer of x digits, and a Fx.y is a real number of length x with y digits to the right of the decimal. For example, an I2 could be a 25, an F6.1 could be 3487.4. A format containing these specifications could be I2, 1X, F6.1. The data card would then look like 25 3487.4, with the 25 beginning in card column one.

card column 1 2 3 4 1234567890123456789012345678901234567890	data element number
1 28943	1
03 03	2
0000.0 2500.0 0500.0 4000.0	3
3.0 3.0 3.0	4
1 2	5
2500.0 7300.0 2400.0 5900.0 2450.0 5300.0	6
02 4900.0 7200.0 7100.0 6700.0 02 5350.0 6000.0 7000.0 6300.0 01 6900.0 5500.0	7
5700.0 7800.0 3.0 190 120 5800.0 6500.0 3.0 180 120 5900.0 5900.0 3.0 180 120	8
0 0500.0 4	9
7000.0 7200.0 7100.0 6500.0 7100.0 5800.0	10
0.85 0.85 0.75 0.70 0.80 0.80 0.75 0.70 0.75 0.75 0.70 0.65 0.60 0.65 0.60 0.55 0.45 0.50 0.50 0.35 0.20 0.20 0.20 0.20	11

Figure 3. Sample Data Deck

card column	1 3 4 1234567890123456789012345678901234567890	data element number
	0.60 0.70 0.65 0.85	
	0.85 0.90 0.85 0.90	
	0.80 0.85 0.85 0.80	11
	0.75 0.80 0.75 0.70	
	0.60 0.70 0.65 0.65	
	0.40 0.45 0.40 0.50	

Figure 3. Sample Data Deck (cont)

V. REQUIRED JCL

The model and line-of-sight data are residing on permanent disk in the W. R. Church Computer Center. In order to exercise the model, the required job control language (JCL) is illustrated in Figure 4. Due to the CPU time required for execution, a time parameter of TIME = 2 must be used on the job card.

```
//JOB CARD , TIME=2
// EXEC FORTCLG
//FORT.SYSIN DD UNIT=2314,VOL=SER=PAT002,DISP=SHR,
// DSN=S1360.SMOLER.PGM,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200)
/*
//GO.FT08F001 DD UNIT=2314,VOL=SER=PAT002,DISP=SHR,
// DSN=S1360.LOSDATA,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200)
/*
//GO.SYSIN DD *

            USER DATA DECK

/*
```

Figure 4. Required JCL

VI. EXPECTED OUTPUT

Once the model has been executed, the user can expect the following output:

1 - Program listing of the model.

2 - A summary of the initial battle conditions, including starting locations and options selected.

3 - A summary of battle conditions after each 10 second time step, including unit locations, force levels, unit status, percentage lost and targets on each unit's target list. The status will show one of the following:

    0 - Unit alive, not firing

    1 - Unit alive and firing

    2 - Unit killed

    3 - Unit moving

4 - A statement of the reason for battle termination.

A sample of the initial battle summary is at Figure 5 and the time step summary is at Figure 6.

INITIAL BATTLE INFORMATION

UNIT	X	Y	FORCE LEVEL
1	1900.0	7800.0	6.0
2	1700.0	6500.0	6.0
3	1700.0	4900.0	6.0
4	5800.0	7700.0	3.0
5	7400.0	6300.0	3.0
6	6200.0	5150.0	3.0

ATTRITION IS STOCHASTIC

ROUTES DETERMINED BY USER

RED VEHICLE SPEED IS 15.0

BREAKPOINT DISTANCE IS 1500.0

BLUE WILL MOVE TO ALTERNATE POSITIONS
ALTERNATE POSITIONS ARE:

UNIT	X	Y
4	9300.0	7300.0
5	9800.0	6200.0
6	9200.0	5800.0

RED KILL PROBABILITIES

RANGE	P	PHH	PHM	PKH
500	0.95	0.95	0.95	1.00
1000	0.80	0.83	0.81	0.95
1500	0.65	0.70	0.70	0.90
2000	0.40	0.50	0.55	0.75
2500	0.30	0.40	0.40	0.75
3000	0.20	0.30	0.25	0.60

BLUE KILL PROBABILITIES

RANGE	P	PHH	PHM	PKH
500	0.80	0.80	0.80	1.00
1000	0.80	0.80	0.80	0.95
1500	0.70	0.70	0.70	0.90
2000	0.65	0.65	0.65	0.85
2500	0.45	0.55	0.45	0.70
3000	0.45	0.55	0.45	0.60

Figure 5. Initial Battle Summary

TIME = 480 SECONDS

UNIT	X	Y	FORCE LEVEL	STATUS	LOST-PCT	TARGETS
1	4933.2	7193.2	5.0	0	0.160	
2	4832.3	5799.9	4.7	0	0.214	
3	4902.6	4579.8	6.0	0	0.0	
4	5800.0	7700.0	0.0	2	1.000	
5	7400.0	6300.0	3.0	1	0.0	2
6	6200.0	5150.0	3.0	3	0.0	

Figure 6. Time Step Summary

APPENDIX B

Program Listing

```
      COMMON /GRP1/ IPRDIR(6),ISECHD(6),MVTDIR(6),X(6),Y(6),SPD(6)
      COMMON /GRP2/ TA(2),TI(2),TH(2),TM(2),TF1(2),TF2(2),TF3(2),
     1P(2,6),PMM(2,6),PHM(2,6),PKH(2,6),TF(2)
      COMMON /GRP3/ NBU,NRU,FL(6),FO(6),NOI(3),XIC(3,200),YIC(3,200),
     1IDIR(3,200),AVSP,ISPD
     1,IUSTAT(6),II(6),LOST(6,6),VISFRA,VISFRB,SIZETK,
     1SIZETH,NT(6),NF(6),SRF,DISMAX,
     1NLOSC(6,6),VISFR(6,6),RMINTK,RMXTK,RMINTH,RMXTH,OP,TOWFR,TNKFR,
     1PTT(3,3),RF,POA(6,6),APOA(6,6),LOA(6,6),NA(6),OFL(6),POL(6)
      COMMON /GRP4/ TPOL(6),OLDQ(6,6),Q(6,6)
      COMMON /GRP5/ LOT(6,6),AOT(6,6)
      COMMON /HILLS/ XC(100),YC(100),PEAK(100),SX(100),SY(100),RHO(100)
      COMMON /HILLS/ SCALE(100),TWORHO(100),TWOSCL(100),BASE
      COMMON /HILLS/ NHILLS
      COMMON /COVER/ CXC(150),CYC(150),CPEAK(150),CPXX(150),CPYY(150)
      COMMON /COVER/ CPXY(150),NCVELS
      COMMON /COUNTR/ KH,KHH,KV,KN,KGRS,KELL,KINT
      COMMON /GRID/ LST(10,10),NHL(10,10),LISTH(450),KHREP(100),KTREP
      COMMON /GRID/ LSTC(10,10),NC(10,10),LISTC(400),KCREP(150)
      COMMON /GRP6/ ALPHA(6)
      COMMON /GRP7/ XA(6),YA(6),IMOVE(6)
C
C     INITIALIZATION.
C
      CALL OVFLOW
      BL=0.0
      AL=0.0
      MP=0
      PAI=22.0/7.0
      ZL=.00001
      CALL SETUP
C
C   READ TERRAIN DATA FOR LINE OF SIGHT
C  CHECK FOR STOCHASTIC OR DETERMINISTIC ATTRITION
C
      READ(5,130) ITRIT,IS
  130 FORMAT(I1,1X,I5)
      DO 132 I=1,6
      CALL RANDOM(IS,YRAN,1)
      ALPHA(I)=(-2.*YRAN**2)+(2.*YRAN+.3)
  132 CONTINUE
C
C  READ IN NUMBER OF BLUE AND RED UNITS
C
      READ(5,200) NBU,NRU
  200 FORMAT(I2,1X,I2)
C
C  INITIALIZE WEAPON SIZES
C
      SIZETK=2.5
```

```
      SIZETW=2.5
C
C  READ IN EFFECTIVE WEAPON RANGES
C
      READ(5,102) RMINTK,RMXTK,RMINTW,RMXTW
  102 FORMAT(F6.1,1X,F6.1,1X,F6.1,1X,F6.1,1X)
C
C  INITIALIZE PM,RF,TOWFR,TNKFR AND NOD
C
      PM=.352
      RF=.5
      TOWFR=.03
      TNKFR=.1
      NOD=2
      DO 101 I=1,NRU
      NOI(I)=125
  101 CONTINUE
      K=NRU+1
      L=NRU+NBU
      DO 111 I=1,L
      II(I)=0
  111 CONTINUE
C
C  READ IN FORCE LEVELS OF EACH RED UNIT
C
      READ(5,103) (FL(I),I=1,NRU)
  103 FORMAT(3(F3.1,1X))
C
C  CHECK FOR TYPE OF ROUTE DETERMINITION
C
      READ(5,106) IRTE,ISPD
  106 FORMAT(I1,1X,I1)
      IF(ISPD.EQ.1) AVSP=9.0
      IF(ISPD.EQ.1) DST=40.232
      IF(ISPD.EQ.2) AVSP=12.0
      IF(ISPD.EQ.2) DST=53.643
      IF(ISPD.EQ.3) AVSP=15.0
      IF(ISPD.EQ.3) DST=67.053
      IF(ISPD.EQ.4) AVSP=18.0
      IF(ISPD.EQ.4) DST=80.463
C
C  READ IN INITIAL RED LOCATIONS
C
      DO 6 I=1,NRU
      READ(5,107) XIC(I,1),YIC(I,1)
  107 FORMAT(F6.1,1X,F6.1)
    6 CONTINUE
      IF(IRTE.EQ.1) GO TO 108
      DO 2 I=1,NRU
      DO 2 J=2,125
```

```
      YIC(I,J)=YIC(I,J-1)+DST*(J-1)
      XIC(I,J)=XIC(I,J-1)+DST*(J-1)
      IDIR(I,J)=0
    2 CONTINUE
      GO TO 109
  108 CALL ROUTE
  109 SUMRO=0.0
      DO 3 I=1,NRU
      FO(I)=FL(I)
      SUMRO=SUMRO+FO(I)
      X(I)=XIC(I,1)
      Y(I)=YIC(I,1)
      MVTDIR(I)=IDIR(I,1)
      SPD(I)=AVSP
      IUSTAT(I)=0
      IPRDIR(I)=IDIR(I,1)
      ISECND(I)=120
      NF(I)=1
      II(I)=1
    3 CONTINUE
C
C   READ IN INITIAL BLUE LOCATIONS
C
      SUMBO=0.0
      DO 4 I=K,L
      READ(5,104) X(I),Y(I),FL(I),IPRDIR(I),ISECND(I)
  104 FORMAT(F6.1,1X,F6.1,1X,F3.1,1X,I3,1X,I3)
      FO(I)=FL(I)
      SUMBO=SUMBO+FO(I)
      MVTDIR(I)=0
      SPD(I)=0.0
      IUSTAT(I)=0
      IMOVE(I)=0
    4 CONTINUE
C
C   CHECK FOR ALTERNATE BLUE POSITIONS AND READ IN IF WANTED
C
      READ(5,400) IALT,BREAK,ITEM
  400 FORMAT(I1,1X,F6.1,1X,I2)
      IF(IALT.EQ.1) GO TO 401
      DO 402 I=K,L
      READ(5,107) XA(I),YA(I)
  402 CONTINUE
  401 DELT=10.
      TA(1)=20.
      TI(1)=8.
      TH(1)=8.
      TM(1)=10.
      TF1(1)=1.
      TF2(1)=1.
```

```
      TF3(1)=1.
      TA(2)=20.
      TI(2)=8.
      TH(2)=8.
      TM(2)=15.
      TF1(2)=10.
      TF2(2)=12.
      TF3(2)=15.
C
C   READ IN HIT AND KILL PROBABILITIES
C
      DO 5 I=1,2
      DO 514 J=1,6
      READ(5,515) P(I,J),PHH(I,J),PHM(I,J),PKH(I,J)
  515 FORMAT(4(F4.2,1X))
  514 CONTINUE
    5 CONTINUE
      PTT(1,1)=1.0
      PTT(1,2)=0.8
      PTT(2,2)=0.2
      PTT(1,3)=0.8
      PTT(2,3)=0.15
      PTT(3,3)=0.05
      DO 31 I=1,NRU
      DO 31 J=K,L
      NLOSC(I,J)=0
      NLOSC(J,I)=0
      Q(I,J)=1.0
      Q(J,I)=1.0
      VISFR(I,J)=0.0
      VISFR(J,I)=0.0
   31 CONTINUE
      IC=1
C
C  PRINT INITIAL BATTLE INFORMATION
C
      WRITE(6,599)
  599 FORMAT('1',1X,'INITIAL BATTLE INFORMATION')
      WRITE(6,600)
  600 FORMAT(/1X,'UNIT',7X,'X',8X,'Y',4X,'FORCE LEVEL')
      DO 601 I=1,L
      WRITE(6,602) I,X(I),Y(I),FL(I)
  602 FORMAT(1X,I3,3X,F7.1,2X,F7.1,7X,F3.1)
  601 CONTINUE
      IF(ITRIT.EQ.1) GO TO 603
      WRITE(6,604)
  604 FORMAT(/1X,'ATTRITION IS STOCHASTIC'/)
      GO TO 605
  603 WRITE(6,606)
  606 FORMAT(/1X,'ATTRITION IS DETERMINISTIC'/)
```

```
  605 IF(IRATE.EQ.0) GO TO 607
      WRITE(6,608)
  608 FORMAT(1X,'ROUTES DETERMINED BY USER'/)
  607 WRITE(6,609) AVSP
  609 FORMAT(1X,'RED VEHICLE SPEED IS ',F4.1/)
      WRITE(6,610) BREAK
  610 FORMAT(1X,'BREAKPOINT DISTANCE IS ',F6.1/)
      IF(IALT.EQ.0) GO TO 615
      WRITE(6,620)
  620 FORMAT(1X,'BLUE WILL NOT MOVE TO ALTERNATE POSITIONS'/)
      GO TO 625
  615 WRITE(6,630)
  630 FORMAT(1X,'BLUE WILL MOVE TO ALTERNATE POSITIONS'/1X,
     1'ALTERNATE POSITIONS ARE:'/1X,'UNIT',5X,'X',8X,'Y')
      DO 635 I=K,L
      WRITE(6,640) I,XA(I),YA(I)
  640 FORMAT(1X,I3,3X,F7.1,2X,F7.1)
  635 CONTINUE
  625 IRAN=500
      WRITE(6,645)
  645 FORMAT(/4X,'RED KILL PROBABILITIES'/1X,'RANGE',4X,'P',
     14X,'PHH',3X,'PHM',3X,'PKH')
      DO 650 I=1,6
      WRITE(6,655) IRAN,P(1,I),PHH(1,I),PHM(1,I),PKH(1,I)
  655 FORMAT(2X,I4,4(2X,F4.2))
      IRAN=IRAN+500
  650 CONTINUE
      IRAN=500
      WRITE(6,660)
  660 FORMAT(/4X,'BLUE KILL PROBABILITIES'/1X,'RANGE',4X,'P',
     14X,'PHH',3X,'PHM',3X,'PKH')
      DO 665 I=1,6
      WRITE(6,655) IRAN,P(2,I),PHH(2,I),PHM(2,I),PKH(2,I)
      IRAN=IRAN+500
  665 CONTINUE
      WRITE(6,670)
  670 FORMAT('1',10X,'BATTLE BEGINS'//)
C
C     UPDATE LOCATION OF RED UNITS.
C
      DISMAX=5000.0
   67 DO 9 I=1,NRU
      IF(IUSTAT(I).EQ.2) GOTO 9
      IF(IUSTAT(I).EQ.0) GOTO 76
      NF(I)=NF(I)+1
      IF(NF(I).LT.NOD) GOTO 9
      NF(I)=1
   76 DO 11 J = 1, NRU
      IF (J .EQ. I) GO TO 11
      IF (IUSTAT(J) .EQ. 2) GO TO 11
```

```
      DIST = X(I) - X(J)
      IF (DIST .GT. DISMAX) GO TO 9
   11 CONTINUE
      II(I) = II(I) + 1
      K7=II(I)
      X(I)=XIC(I,K7)
      Y(I)=YIC(I,K7)
      MVTDIR(I)=IDIR(I,K7)
      IPRDIR(I)=IDIR(I,K7)
    9 CONTINUE
C
C     LINE--OF-SIGHT CHECK BETWEEN UNITS AND TARGETS SELECTION
C
      DO 17 J=K,L
      NT(J)=0
   17 CONTINUE
      DO 12 I=1,NRU
      NT(I)=0
      IF(IUSTAT(I).EQ.2) GOTO 12
      DO 16 J=K,L
      IF(IUSTAT(J).EQ.2.OR.IUSTAT(J).EQ.3) GO TO 16
      XX1=X(I)
      YY1=Y(I)
  133 CALL ELEV(XX1,YY1,TMACI)
      XX2=X(J)
      YY2=Y(J)
      CALL ELEV(XX2,YY2,TMACJ)
      LATOB=1
      LBTOA=1
      CALL LOS(XX1,YY1,TMACI,0.0,SIZETK,XX2,YY2,TMACJ,0.0,SIZETW,
     1LATOB,LBTOA,VISFRA,VISFRB)
      VISFR(I,J)=VISFRA
      VISFR(J,I)=VISFRB
      IF(VISFRA.GT.ZL) GOTO 18
      LOST(I,J)=0
      LOST(J,I)=0
      NLOSC(I,J)=NLOSC(I,J)+1
      NLOSC(J,I)=NLOSC(I,J)
      GOTO 16
   18 LOST(I,J)=1
      LOST(J,I)=1
      NLOSC(I,J)=0
      NLOSC(J,I)=0
      RANGE=SQRT((X(I)-X(J))**2+(Y(I)-Y(J))**2)
      IF(RANGE.LT.RMINTK.OR.RANGE.GT.RMXTK)GOTO 20
      IF(Q(I,J).EQ.1.0) GOTO 20
      IUSTAT(I)=1
      NT(I)=NT(I)+1
      M=NT(I)
      LOT(I,M)=J
```

```
      ROT(I,M)=RANGE
      IF(M.EQ.1) GOTO 20
      CALL SORT(I,M)
   20 IF(RANGE.LT.RMINTW.OR.RANGE.GT.RMXTW) GOTO 16
      IF(Q(J,I).EQ.1.0) GOTO 16
      IUSTAT(J)=1
      NT(J)=NT(J)+1
      M=NT(J)
      LOT(J,M)=I
      ROT(J,M)=RANGE
      IF(M.EQ.1) GOTO 16
      CALL SORT(J,M)
   16 CONTINUE
   12 CONTINUE
      DO 25 I=1,NRU
      IF(IUSTAT(I).EQ.2) GOTO 25
      IF(NT(I).NE.0) GOTO 25
      IUSTAT(I)=0
      NF(I)=0
   25 CONTINUE
      DO 79 J=K,L
      IF(IUSTAT(J).EQ.2.OR.IUSTAT(J).EQ.3) GO TO 79
      IF(NT(J).EQ.0) IUSTAT(J)=0
   79 CONTINUE
C
C     UPDATE OF THE ACCUMULATED DETECTION PROBABILITIES.
C
      IAA=1
      IBB=NRU
      ICC=K
      IDD=L
      FR=TOWFR
      OP=PM
   37 DO 14 I=IAA,IBB
      IF(IUSTAT(I).EQ.2.OR.IUSTAT(I).EQ.3) GO TO 14
      DO 19 J=ICC,IDD
      PROP=0.0
      IF(IUSTAT(J).EQ.2.OR.IUSTAT(J).EQ.3) GO TO 19
      OLDQ(I,J)=Q(I,J)
      IF(LOST(I,J).EQ.0) GOTO 15
      IF(NT(I).GT.0) GOTO 22
      PCTVIS=VISFR(J,I)
      CALL LAMDA(I,J,PCTVIS,DETRAT,PSUBK)
      QV=EXP(-(FL(I)*DETRAT*OP*DELT*FL(J)))
      IF(NT(J).GT.0) GOTO 23
      Q(I,J)=Q(I,J)*QV
      GOTO 19
   23 QP=(1.0-PSUBK)**(FR*DELT*FL(I)*FL(J))
      Q(I,J)=Q(I,J)*(QV+QP-QV*QP)
      GOTO 19
```

```
      22 N5=NT(I)
         DO 24 I1=1,N5
         K1=LOT(I,I1)
         ANG1=ATAN2(Y(K1)-Y(I),X(K1)-X(I))
         ANG2=ATAN2(Y(J)-Y(I),X(J)-X(I))
         IF((ANG1*ANG2).GE.0.0) GOTO 77
         IF(ANG2.LT.0.0) GOTO 32
         ANG=2*PAI+ANG1-ANG2
         GOTO 35
      32 ANG=2*PAI+ANG2-ANG1
      35 IF(ANG.GT.PAI) ANG=2*PAI-ANG
         GOTO 33
      77 ANG=ABS(ANG2-ANG1)
      33 AA=15.0*PAI/180.0
         IF(ANG.GT.AA) GOTO 24
         PROP=PROP+PTT(I1,N5)
      24 CONTINUE
         IF(PROP.EQ.0.0) GOTO 34
         IF(NT(J).GT.0)GOTO 36
         CALL LAMDA(I,J,PCTVIS,DETRAT,PSUBK)
         DETRAT=DETRAT*AF
         QV=EXP(-(FL(I)*PROP*DETRAT*DELT*FL(J)))
         Q(I,J)=Q(I,J)*QV
         GOTO 19
      36 Q(I,J)=0.0
         GOTO 19
      34 IF(IAA.EQ.1) GOTO 19
         Q(I,J)=1.0
         GOTO 19
      15 IF(NLOSC(I,J).LE.3) GOTO 19
         Q(I,J)=1.0
      19 CONTINUE
      14 CONTINUE
         IF(IAA.EQ.K) GOTO 38
         FA=TNKFA
         IAA=K
         IBB=L
         ICC=1
         IDD=NAU
         OP=1.0
         GOTO 37
C
C        FIRE ALLOCATION.
C
      38 DO 28 I=1,L
      28 NA(I)=0
         DO 26 I=1,L
         IF(IUSTAT(I).EQ.2.OR.IUSTAT(I).EQ.3) GO TO 26
         IF(NT(I).EQ.0) GOTO 26
         DO 27 J=1,3
```

```
      APOA(I,J)=0.0
   27 CONTINUE
      IF(NT(I).EQ.1) GOTO 78
      IF(NT(I).EQ.2) GOTO 29
      NOT=3
      MM1=LOT(I,1)
      MM2=LOT(I,2)
      MM3=LOT(I,3)
      PROB=(1.0-Q(I,MM1))*Q(I,MM2)*Q(I,MM3)
      APOA(I,1)=APOA(I,1)+PTT(1,1)*PROB
      PROB=Q(I,MM1)*(1.0-Q(I,MM2))*Q(I,MM3)
      APOA(I,2)=APOA(I,2)+PTT(1,1)*PROB
      PROB=Q(I,MM1)*Q(I,MM2)*(1.0-Q(I,MM3))
      APOA(I,3)=APOA(I,3)+PTT(1,1)*PROB
      PROB=(1.0-Q(I,MM1))*(1.0-Q(I,MM2))*Q(I,MM3)
      APOA(I,1)=APOA(I,1)+PTT(1,2)*PROB
      APOA(I,2)=APOA(I,2)+PTT(2,2)*PROB
      PROB=(1.0-Q(I,MM1))*Q(I,MM2)*(1.0-Q(I,MM3))
      APOA(I,1)=APOA(I,1)+PTT(1,2)*PROB
      APOA(I,3)=APOA(I,3)+PTT(2,2)*PROB
      PROB=Q(I,MM1)*(1.0-Q(I,MM2))*(1.0-Q(I,MM3))
      APOA(I,2)=APOA(I,2)+PTT(1,2)*PROB
      APOA(I,3)=APOA(I,3)+PTT(2,2)*PROB
      PROB=(1.0-Q(I,MM1))*(1.0-Q(I,MM2))*(1.0-Q(I,MM3))
      APOA(I,1)=APOA(I,1)+PTT(1,3)*PROB
      APOA(I,2)=APOA(I,2)+PTT(2,3)*PROB
      APOA(I,3)=APOA(I,3)+PTT(3,3)*PROB
   30 DO 44 J=1,NOT
      KK=LOT(I,J)
      NA(KK)=NA(KK)+1
      IN=NA(KK)
      LOA(KK,IN)=I
      POA(KK,IN)=APOA(I,J)
   44 CONTINUE
      GOTO 26
   29 NOT=2
      MM1=LOT(I,1)
      MM2=LOT(I,2)
      PROB=(1.0-Q(I,MM1))*Q(I,MM2)
      APOA(I,1)=APOA(I,1)+PTT(1,1)*PROB
      PROB=Q(I,MM1)*(1.0-Q(I,MM2))
      APOA(I,2)=APOA(I,2)+PTT(1,1)*PROB
      PROB=(1.0-Q(I,MM1))*(1.0-Q(I,MM2))
      APOA(I,1)=APOA(I,1)+PTT(1,2)*PROB
      APOA(I,2)=APOA(I,2)+PTT(2,2)*PROB
      GOTO 30
   78 NOT=1
      MM1=LOT(I,1)
      PROB=1.0-Q(I,MM1)
      APOA(I,1)=APOA(I,1)+PTT(1,1)*PROB
```

```
      GOTO 30
   26 CONTINUE
C
C     ATTRITION.
C
      SUMA=0.0
      SUMB=0.0
      DO 40 I=1,L
      IF(IUSTAT(I).EQ.2.OR.IUSTAT(I).EQ.3) GO TO 40
      M6=NA(I)
      SUM=0.0
      IF(M6.EQ.0) GOTO 47
      DO 41 J=1,M6
      M7=LOA(I,J)
      IF(M7.LT.K) GOTO 42
      ITYPE=2
      GOTO 43
   42 ITYPE=1
   43 RANGE=SQRT((X(I)-X(M7))**2+(Y(I)-Y(M7))**2)
      IF (ITRIT.EQ.1) GO TO 131
      CALL STOCH(ITYPE,RANGE,AJI)
      GO TO 5000
  131 CALL ETK(ITYPE,RANGE,T)
      AJI=1.0/T
 5000    SUM=SUM+AJI*FL(M7)*POA(I,J)*DELT
   41 CONTINUE
   47 OFL(I)=FL(I)
      FL(I)=FL(I)-SUM
      IF(FL(I).GT.ZL) GOTO 46
      FL(I)=0.0
      IUSTAT(I)=2
   46 IF(I.LT.K) GOTO 60
      SUMB=SUMB+FL(I)
      TPOL(I)=(FO(I)-FL(I))/FO(I)
      GO TO 40
   60 SUMA=SUMA+FL(I)
      TPOL(I)=(FO(I)-FL(I))/FO(I)
   40 CONTINUE
C
C     PRINT AND CHECK FOR BATTLE DETERMINATION.
C
      ITIME=IC*10
      DO 57 I=K,L
      IF(IUSTAT(I).EQ.2) GO TO 57
      DO 58 J=1,NAU
      IF(IUSTAT(J).EQ.2) GO TO 58
      CHECK=X(I)-X(J)
      AVD=SQRT((X(I)-X(J))**2+(Y(I)-Y(J))**2)
      IF(AVD.LT.BREAK.OR.CHECK.LT.50.) GO TO 250
   58 CONTINUE
```

```
   57 CONTINUE
      GO TO 99
C
C   COMPLETE BLUE MOVE
C
  250 DO 251 I=K,L
      IF(IALT.EQ.1.OR.IMOVE(I).EQ.ITEM) GO TO 6000
      IF(IUSTAT(I).EQ.0) IUSTAT(I)=3
      IMOVE(I)=IMOVE(I)+1
      IF(IMOVE(I).LT.ITEM) GO TO 251
      X(I)=XA(I)
      Y(I)=YA(I)
      IF(IUSTAT(I).EQ.3) IUSTAT(I)=0
  251 CONTINUE
   99 WRITE(6,112) ITIME
  112 FORMAT(////1X,'TIME = ',I4,1X,'SECONDS'//)
      WRITE(6,113)
  113 FORMAT(1X,'UNIT',5X,'X',8X,'Y',5X,'FORCE LEVEL',2X,'STATUS',
     12X,'LOST-PCT',2X,'TARGETS')
      DO 59 I=1,L
      N6=NT(I)
      IF(N6.NE.0) GO TO 48
      WRITE(6,264) I,X(I),Y(I),FL(I),IUSTAT(I),TPOL(I)
  264 FORMAT(3X,I1,3X,F7.1,2X,F7.1,6X,F3.1,9X,I1,6X,F5.3)
      GO TO 59
   48 WRITE(6,114) I,X(I),Y(I),FL(I),IUSTAT(I),TPOL(I),
     1(LOT(I,J),J=1,N6)
  114 FORMAT(3X,I1,3X,F7.1,2X,F7.1,6X,F3.1,9X,I1,6X,F5.3,3X,3(I1,1X))
   59 CONTINUE
C
C     CHECK FOR BATTLE DETERMINATION.
C
      IOT=0
      DO 53 I=1,NRU
      IF(FL(I).EQ.0.0) GOTO 53
      IOT=1
   53 CONTINUE
      IF(IOT.EQ.1) GOTO 54
      WRITE(6,117)
  117 FORMAT(1X,'*** RED FORCE IS ELIMINATED.  END OF BATTLE.')
      GOTO 66
   54 IOT=0
      DO 55 I=K,L
      IF(FL(I).EQ.0.0) GOTO 55
      IOT=1
   55 CONTINUE
      IF(IOT.EQ.1) GOTO 65
      WRITE(6,118)
  118 FORMAT(1X,'***  BLUE FORCE IS ELIMINATED. END OF BATTLE.')
      GOTO 66
```

```
 6000 WRITE (6,119)
  119 FORMAT(1X,'*** DISTANCE BETWEEN FORCES IS TOO CLOSE. END OF BATTLE
     1')
      GOTO 66
   65 IC=IC+1
      GOTO 67
   66 STOP
      END
```

```
      SUBROUTINE LOS(XA,YA,TMACA,THICA,SIZEA,XB,YB,TMACB,THICB,SIZEB,
     -LATOB,LOTOA,VISFRA,VISFRB)
C
      COMMON /HILLS/ XC(100),YC(100),PEAK(100),SX(100),SY(100),RHO(100)
      COMMON /HILLS/ SCALE(100),TWORHO(100),TWOSCL(100),BASE
      COMMON /HILLS/ NHILLS
      COMMON /COVER/ CXC(150),CYC(150),CPEAK(150),CPXX(150),CPYY(150)
      COMMON /COVER/ CPXY(150),NCVELS
      COMMON /COUNTR/KH, KHH, KV, KN, KGRS, KELL, KINT
      COMMON /GRID/ LST(10,10),NHL(10,10),LISTH(450),KHREP(100),KTREP
      COMMON /GRID/ LSTC(10,10),NC(10,10),LISTC(400),KCREP(150)
      DIMENSION IGX(100),IGY(100),IEL(100),CS1(100),CS2(100)
      DATA NGRID/10/,GSIZE/1000./
C SUBROUTINE TO COMPUTE FRACTION VISIBLE FOR OBSERVER TARGET PAIRS
      VISFRA=1.
      VISFRB=1.
      XBA=XB-XA
      YBA=YB-YA
      IF((XBA.EQ.0.).AND.(YBA.EQ.0.)) RETURN
      IF(SIZEA+THICA.LE.0.) GO TO 510
      IF(SIZEB+THICB.LE.0.) GO TO 510
      IF(THICA.LT.0.) VISFRA=1.0+THICA/SIZEA
      IF(THICB.LT.0.) VISFRB=1.0+THICB/SIZEB
      ZA=TMACA + THICA + SIZEA
      ZB=TMACB + THICB + SIZEB
      KTREP=KTREP+1
      ZBA=ZB-ZA
      XBASQ=XBA*XBA
      YBASQ=YBA*YBA
      XYBA=XBA*YBA
      TWOXBA=2.*XBA
      TWOYBA=2.*YBA
C COMPUTE GRID SQUARES CROSSED BY A TO B LINE
      NGRSQ=0
      IF(XBA) 110,95,100
95    XBA=0.1
100   ISGX=-1
      XINC=GSIZE/XBA
      GO TO 120
110   ISGX=1
      XINC=-GSIZE/XBA
120   IF(YBA)  140,125,130
125   YBA=0.1
130   ISGY=-1
      YINC=GSIZE/YBA
      GO TO 150
140   ISGY=1
      YINC=-GSIZE/YBA
150   IX=1+IFIX(XB/GSIZE)
      IF(IX.GT.NGRID) IX=NGRID
```

```
      IY=1+IFIX(YB/GSIZE)
      IF(IY.GT.NGRID) IY=NGRID
      XNEXT=GSIZE*(FLOAT(IX)+0.5*(ISGX-1.))
      YNEXT=GSIZE*(FLOAT(IY)+0.5*(ISGY-1.))
      XSTEP=(XB-XNEXT)/XBA
      YSTEP=(YB-YNEXT)/YBA
160   NGRSQ=NGRSQ+1
      IGX(NGRSQ)=IX
      IGY(NGRSQ)=IY
      IF((XSTEP.GT.1.).AND.(YSTEP.GT.1.)) GO TO 200
      IF(XSTEP-YSTEP) 170,180,190
170   IX=IX +ISGX
      XSTEP=XSTEP+XINC
      GO TO 160
180   IX=IX+ISGX
      XSTEP=XSTEP+XINC
190   IY=IY+ISGY
      YSTEP=YSTEP+YINC
      GO TO 160
200   KGRS=KGRS+NGRSQ
C GRID SQUARE LIST NOW COMPLETE IN IGX, IGY WITH NGRSQ ENTRIES
C
C NOW FIND WHICH COVER ELLIPSES TOUCH THE A TO B LINE,
C CHECK ELEVATIONS AT S1 AND S2 FOR EACH SUCH ELLIPSE
      NELS=0
      CHTMAX=0.
      IF(NCVELS.EQ.0) GOTO 270
      DO 260 K=1,NGRSQ
      IX=IGX(K)
      IY=IGY(K)
      N=NC(IX,IY)
      IF(N.EQ.0) GO TO 260
      LS=LSTC(IX,IY)
      LEND=LS+N-1
      DO 250 L=LS,LEND
      KELL=KELL+1
      IC=LISTC(L)
      IF(KCREP(IC).EQ.KTREP) GO TO 250
      KCREP(IC)=KTREP
      AX=XA-CXC(IC)
      AY=YA-CYC(IC)
      PXX=CPXX(IC)
      PYY=CPYY(IC)
      PXY=CPXY(IC)
      AA=PXX*XBASQ+PYY*YBASQ+PXY*XYBA
      BB=PXX*TWOXBA*AX+PYY*TWOYBA*AY+PXY*(AX*YBA+AY*XBA)
      CC=PXX*AX*AX+PYY*AY*AY+PXY*AX*AY-1.0
      ARG=BB*BB-4.0*AA*CC
      IF(ARG.LE.0.) GO TO 250
      SQ=SQRT(ARG)
```

```
      S1=-(BB+SQ)/(2.0*AA)
      S2=(SQ-BB)/(2.0*AA)
      IF(S1.GE.1.) GO TO 250
      IF(S2.LE.0.) GO TO 250
      IF(S1.LE.0.)  GO TO 510
      IF(S2.GE.1.) GO TO 510
C NOW CHECK LOS AT S1 AND S2
      KINT=KINT+1
      CPK=CPEAK(IC)
      XS=XA+S2*XBA
      YS=YA+S2*YBA
      CALL ELEV(XS,YS,HTS)
      HTS=HTS+CPK
      ZS=ZA+S2*ZBA
      IF(LATOB.EQ.0) GO TO 210
      CALL KOVER(ZA,THACB,SIZEB,ZB,S2,HTS,ZS,VISFRB)
      IF(VISFRB.LE.0.) GO TO 510
  210 IF(LBTOA.EQ.0) GO TO 220
      S=1.-S2
      CALL KOVER(ZB,THACA,SIZEA,ZA,S,HTS,ZS,VISFRA)
      IF(VISFRA.LE.0.) GO TO 510
  220 XS=XA+S1*XBA
      YS=YA+S1*YBA
      CALL ELEV(XS,YS,HTS)
      HTS=HTS+CPK
      ZS=ZA+S1*ZBA
      IF(LATOB.EQ.0) GO TO 230
      CALL KOVER(ZA,THACB,SIZEB,ZB,S1,HTS,ZS,VISFRB)
      IF(VISFRB.LE.0.) GO TO 510
  230 IF(LBTOA.EQ.0) GO TO 240
      S=1.0-S1
      CALL KOVER(ZB,THACA,SIZEA,ZA,S,HTS,ZS,VISFRA)
      IF(VISFRA.LE.0.) GO TO 510
  240 NELS=NELS+1
      IEL(NELS)=IC
      CS1(NELS)=S1
      CS2(NELS)=S2
      IF(CPK.GT.CHTMAX) CHTMAX=CPK
  250 CONTINUE
  260 CONTINUE
C ALL ELLIPSES CHECKED
C
C NOW START ON THE HILLS
  270 DO 600 K=1,NGRSQ
      IX=IGX(K)
      IY=IGY(K)
      IF(NHL(IX,IY).EQ.0) GO TO 600
      LS=LST(IX,IY)
      LEND=LS+NHL(IX,IY)-1
      DO 500 L=LS,LEND
```

```
      I=LISTH(L)
      IF(KHREP(I).EQ.KTREP) GO TO 500
      KHREP(I)=KTREP
C  PROCESSING FOR HILL I STARTS HERE
      KH=KH+1
C COMPUTE W =TOP OF THIS HILL ALONG O-T LINE
      CX=XBA/SX(I)
      CY=YBA/SY(I)
      DX=(XA-XC(I))/SX(I)
      DY=(YA-YC(I))/SY(I)
      FQ=TWOSCL(I)*(CX*DX+CY*DY+RHO(I)*(CX*DY+CY*DX))
      GQ=SCALE(I)*(CX*CX+CY*CY+TWORHO(I)*CX*CY)
      IF(GQ .EQ.0.)  GO TO 500
      W=-FQ/(2.*GQ)
      IF(ABS(W).GT.5.) GO TO 500
      FSQ=FQ*FQ
      EQ=SCALE(I)*(DX*DX+DY*DY+TWORHO(I)*DX*DY)
      POWER=EQ-FSQ/(4.*GQ)
      IF(POWER .LT. -3.) GO TO 500
      HHW=PEAK(I)*EXP(POWER)
      KHW=KHW+1
      IF(HHW.LE.BASE) GO TO 500
      ZW=ZA+W*ZBA
      IF((W.LT.0.).OR.(W.GT.1.)) GO TO 300
      IF(HHW .GE. ZW) GO TO 510
      CVHTW=0.
      IF(NELS.EQ.0) GO TO 300
      DO 280 M=1,NELS
      IF((CS1(M).GE.W).OR.(CS2(M).LE.W)) GO TO 280
      IC=IEL(M)
      IF(CVHTW.LT.CPEAK(IC))  CVHTW=CPEAK(IC)
280   CONTINUE
      IF((HHW+CVHTW).GE.ZW)  GO TO 510
300   IF(HHW+CHTMAX.LT.AMIN1(ZA-SIZEA,ZB-SIZEB)) GO TO 500
C IF WE GET TO HERE THEN NEED TO FIND LOWEST SIGHT LINE OVER HILL
C NEWTON ITERATION A TO B GIVING VISFAB
      IF(LATOB.EQ.0) GO TO 400
      KV=KV+1
      V=W
      HHV=HHW
      NCT=0
      FV=FQ*V
      TWOGV=2.*GQ*V
330    FCNV=ZA+HHV*(TWOGV*V+FV-1.)
      KN=KN+1
      FACTOR=(TWOGV*TWOGV+2.*(GQ+TWOGV*FQ)+FSQ)
      DFCNV=HHV*V*FACTOR
      IF(ABS(DFCNV) .LT.1.E-10) GO TO 950
      V=V-FCNV/DFCNV
      IF(ABS(V).GT.5.)GO TO 400
```

```
      FV=FQ*V
      TWOGV=2.*GQ*V
      POWER = EQ+FV+GQ*V*V
      IF(POWER .LT. -3.) GO TO 400
      HHV=PEAK(I)*EXP(POWER)
      DHHV=HHV*(FQ+TWOGV)
      ELV=ZA+DHHV*V
      IF (ABS(HHV-ELV) .LT.1.) GO TO 350
      NCT=NCT+1
      IF (NCT.LT.10) GO TO 330
  350 IF((V.LT.0.).OR.(V.GT.1.)) GO TO 400
      CVHTV=0.
      IF(NELS.EQ.0) GO TO 390
      DO 380 M=1,NELS
      IF((CS1(M).GE.V).OR.(CS2(M).LE.V))GO TO 380
      IC=IEL(M)
      IF(CVHTV.LT.CPEAK(IC)) CVHTV=CPEAK(IC)
  380 CONTINUE
  390 HTV=HHV+CVHTV
      ZV=ZA+V*ZBA
      CALL KOVER(ZA,THACB,SIZEB,ZB,V,HTV,ZV,VISFRB)
      IF (VISFRB.LE.0.) GO TO 510
C NEWTON ITERATION B TO A GIVING VISFRA
  400 IF(LBTOA.EQ.0) GO TO 500
      KV=KV+1
      V=W
      VM1=V-1.
      HHV=HHW
      NCT=0
      FV=FQ*V
      TWOGV=2.*GQ*V
  430 FCNV=ZB+HHV*((FQ+TWOGV)*VM1-1.)
      KN=KN+1
      FACTOR=(TWOGV*TWOGV+2.*(GQ+TWOGV*FQ)+FSQ)
      DFCNV=HHV*VM1*FACTOR
      IF(ABS(DFCNV) .LT.1.E-10) GO TO 450
      V=V-FCNV/DFCNV
      IF(ABS(V).GT.5.)GO TO 500
      VM1=V-1.
      FV=FQ*V
      TWOGV=2.*GQ*V
      POWER = EQ+FV+GQ*V*V
      IF(POWER .LT. -3.) GO TO 500
      HHV=PEAK(I)*EXP(POWER)
      DHHV=HHV*(FQ+TWOGV)
      ELV=ZB+DHHV*VM1
      IF (ABS(HHV-ELV) .LT.1.) GO TO 450
      NCT=NCT+1
      IF (NCT.LT.10) GO TO 430
  450 IF((V.LT.0.).OR.(V.GT.1.)) GO TO 500
```

```
      CVHTV=0.
      IF(NELS.EQ.0) GO TO 490
      DO 480 M=1,NELS
      IF((CS1(M).GE.V).OR.(CS2(M).LE.V))GO TO 480
      IC=IEL(M)
      IF(CVHTV.LT.CPEAK(IC)) CVHTV=CPEAK(IC)
  480 CONTINUE
  490 HTV=HHV+CVHTV
      ZV=ZA+V*ZBA
      S=-VM1
      CALL KOVER(ZB,THACA,SIZEA,ZA,S,HTV,ZV,VISFRA)
      IF (VISFRA.LE.0.) GO TO 510
  500 CONTINUE
  600 CONTINUE
      RETURN
  510 VISFRA=0.
      VISFRB=0.
      RETURN
      END
```

```
      SUBROUTINE SETUP
C
      COMMON /HILLS/ XC(100),YC(100),PEAK(100),SX(100),SY(100),RHO(100)
      COMMON /HILLS/ SCALE(100),TWORHO(100),TWOSCL(100),BASE
      COMMON /HILLS/ NHILLS
      COMMON /COVER/ CXC(150),CYC(150),CPEAK(150),CPXX(150),CPYY(150)
      COMMON /COVER/ CPXY(150),NCVELS
      COMMON /COUNTR/KH,KHW,KV,KN,KGRS,KELL,KINT
      COMMON /GRID/ LST(10,10),NHL(10,10),LISTH(450),KHREP(100),KTREP
      COMMON /GRID/ LSTC(10,10),NC(10,10),LISTC(400),KCREP(150)
      READ(8,7) NHILLS
      READ(8,47) BASE
47    FORMAT(F10.4)
7     FORMAT(I6)
17    FORMAT(5F8.3,F6.4)
      DO 50 I=1,NHILLS
      READ(8,17) XC(I),YC(I),PEAK(I),SX(I),SY(I),RHO(I)
50    CONTINUE
      READ(8,37)LST
      READ(8,37)NHL
      READ(8,7)NHTOT
      READ(8,37) (LISTH(I),I=1,NHTOT)
37    FORMAT(16I5)
      READ(8,7)NCVELS
      IF(NCVELS.EQ.0) GO TO 65
      DO 60 I=1,NCVELS
      READ(8,27)CXC(I),CYC(I),CPEAK(I),CPXX(I),CPYY(I),CPXY(I)
27    FORMAT(3F10.4,3E13.7)
      KCREP(I)=-2147483600
60    CONTINUE
      READ(8,37)LSTC
      READ(8,37)NC
      READ(8,7)NCTOT
      READ(8,37) (LISTC(I),I=1,NCTOT)
   65 DO 100 I=1,NHILLS
      SX(I)=SX(I)*1.625
      SY(I)=SY(I)*1.625
      XC(I)=(XC(I)-500.)*100.
      YC(I)=(YC(I)-930.)*100.
      TWORHO(I)=2.*RHO(I)
      SCALE(I)=-1./(2.*(1.-RHO(I)**2))
      TWOSCL(I)=2.*SCALE(I)
      KHREP(I)=-2147483600
C  ALL VALUES NOW IN METERS ON 0 -- 10,000 GRID
100   CONTINUE
      KTREP=-2147483600
      KH=0
      KHW=0
      KV=0
      KN=0
```

```
      KGRS=0
      KELL=0
      KINT=0
      RETURN
      END
```

```
C
      SUBROUTINE ROUTE
C
      COMMON /GRP3/ NBU,NRU,FL(6),FO(6),NOI(3),XIC(3,200),YIC(3,200),
     1IDIR(3,200),AVSP,ISPD
     1,IUSTAT(6),II(6),LOST(6,6),VISFRA,VISFRB,SIZETK,
     1SIZETW,NT(6),NF(6),SRF,DISMAX,
     1NLOSC(6,6),VISFR(6,6),RMINTK,RMXTK,RMINTW,RMXTW,OP,TOWFR,TNKFR,
     1PTT(3,3),RF,POA(6,6),APOA(6,6),LOA(6,6),NA(6),OFL(6),POL(6)
      DIMENSION XLOC(3,20),YLOC(3,20),N(3)
      IF(ISPD.EQ.4) DST=80.463
      IF(ISPD.EQ.3) DST=67.053
      IF(ISPD.EQ.2) DST=53.643
      IF(ISPD.EQ.1) DST=40.232
      DO 300 I=1,NRU
      READ(5,15) N(I)
   15 FORMAT(I2)
      NL=N(I)+1
      DO 200 IN=2,NL
      READ(5,201) XLOCS,YLOCS
  201 FORMAT(F6.1,1X,F6.1)
      XLOC(I,IN)=XLOCS
      YLOC(I,IN)=YLOCS
  200 CONTINUE
      XLOC(I,1)=XIC(I,1)
      YLOC(I,1)=YIC(I,1)
      IDIR(I,1)=0
      NL=N(I)
      NUM=2
      DO 305 J=1,NL
      XL=XLOC(I,J+1)-XLOC(I,J)
      YL=YLOC(I,J+1)-YLOC(I,J)
      DIST=SQRT(XL**2+YL**2)
      Y=ABS(YL)
      Z=Y/XL
      ANGL=ATAN(Z)
      DEG=ANGL*57.2958
      IF(J.EQ.1) GO TO 320
      XLN=(DST-EXTRA)*COS(ANGL)
      DIST=(DIST+EXTRA)-DST
      YLN=(DST-EXTRA)*SIN(ANGL)
      XIC(I,NUM)=XIC(I,NUM-1)+XLN+XLE
      IF(YL.GT.0.) GO TO 325
      YLN=-YLN
  325 YIC(I,NUM)=YIC(I,NUM-1)+YLN+YLE
      IF(YL.GT.0.) GO TO 340
      IDIR(I,NUM)=-IFIX(DEG)
      GO TO 341
  340 IDIR(I,NUM)=IFIX(DEG)
  341 NUM=NUM+1
```

```
  320 XLN=DST*COS(ANGL)
      YLN=DST*SIN(ANGL)
      IF(YL.GT.0.) GO TO 310
      YLN=-YLN
  310 IF(DIST.LT.DST) GO TO 315
      XIC(I,NUM)=XIC(I,NUM-1)+XLN
      YIC(I,NUM)=YIC(I,NUM-1)+YLN
      IF(YL.GT.0.) GO TO 342
      IDIR(I,NUM)=-IFIX(DEG)
      GO TO 343
  342 IDIR(I,NUM)=IFIX(DEG)
  343 DIST=DIST-DST
      NUM=NUM+1
      GO TO 310
  315 EXTRA=DIST
      XLE=EXTRA*COS(ANGL)
      YLE=EXTRA*SIN(ANGL)
      IF(YL.GT.0.) GO TO 305
      YLE=-YLE
  305 CONTINUE
  300 CONTINUE
      RETURN
      END
```

```
C
      SUBROUTINE LAMDA(I,J,PCTVIS,DETRAT,PK)
C
C     SUBROUTINE TO COMPUTE DETECTION RATE (DETRAT) OF TARGET J
C     BY OBSERVER I GIVEN THE VISIBLE FRACTION (PCTVIS).
C
      COMMON /GRP1/ IPRDIR(6),ISECWD(6),MVTDIR(6),X(6),Y(6),SPD(6)
      TCFACT=1.0
      ZEROL=0.00001
      PAI=22.0/7.0
    7 D=(ISECWD(I)*PAI/180.0)/2.0
      BBB=(1.0/(2.0*(SIN(D)-D*COS(D))))
      IF(ABS(BBB).LT.ZEROL) BBB=0.0
      AAA=(-BBB)*COS(D)
      IF(ABS(AAA).LT.ZEROL) AAA=0.0
      OTANG=ATAN2((Y(J)-Y(I)),(X(J)-X(I)))
      PD=IPRDIR(I)*PAI/180.0
      IF((PD*OTANG).GE.0.0) GOTO 1
      IF(PD.LT.0.0) GOTO 9
      ANGLE=2*PAI+OTANG-PD
      GOTO 10
    9 ANGLE=2*PAI+PD-OTANG
   10 IF(ANGLE.GT.PAI) ANGLE=2*PAI-ANGLE
      GOTO 2
    1 ANGLE=ABS(PD-OTANG)
    2 IF(ANGLE.GT.D) GOTO 3
      DUP=PD+D
      DLOW=PD-D
      ANGLFT=OTANG+(15.0*PAI/180.)
      IF(ANGLFT.GT.DUP) ANGLFT=DUP
      ANGLRT=OTANG-(15.*PAI/180.)
      IF(ANGLRT.LT.DLOW) ANGLRT=DLOW
      PK=BBB*(SIN(ANGLFT)-SIN(ANGLRT))+AAA*(ANGLFT-ANGLRT)
      IF(PK.LT.0.0) GOTO 3
      IF(PK.GT.1.0) GOTO 5
      GOTO 8
    3 PK=0.0
      DETRAT=0.0
      GOTO 6
    5 PK=1.0
    8 RANGE=SQRT((X(J)-X(I))**2+(Y(J)-Y(I))**2)
      RR=0.001*RANGE/PCTVIS
      TOANG=ATAN2((Y(I)-Y(J)),(X(I)-X(J)))
      AD=MVTDIR(J)*PAI/180.0
      HORVEL=ABS(SPD(J)*SIN(TOANG-AD))
      DENOM=1.453+TCFACT*(0.5978+2.188*(RR**2)-0.5038*HORVEL)
      IF(DENOM.LE.ZEROL) DENOM=ZEROL
      DETRAT=0.003+1.088/DENOM
      DETRAT=DETRAT*PK
    6 RETURN
```

```
      SUBROUTINE ELEV(X,Y,TMAC)
C
      COMMON /HILLS/ XC(100),YC(100),PEAK(100),SX(100),SY(100),RHO(100)
      COMMON /HILLS/ SCALE(100),TWORHO(100),TWOSCL(100),BASE
      COMMON /HILLS/ NHILLS
      COMMON /GRID/ LST(10,10),NHL(10,10),LISTH(450),KHREP(100),KTREP
      COMMON /GRID/ LSTC(10,10),NC(10,10),LISTC(400),KCREP(150)
      DATA NGRID/10/,GSIZE/1000./
C FUNCTION TO COMPUTE TERRAIN ELEVATION FOR GIVEN X, Y COORDINATES.
      ZMAX=BASE
      IX=1+IFIX(X/GSIZE)
      IF(IX.GT.NGRID)   IX=NGRID
      IY=1+IFIX(Y/GSIZE)
      IF(IY.GT.NGRID)   IY=NGRID
      IF(NHL(IX,IY).EQ.0) GO TO 150
      LS=LST(IX,IY)
      LEND=LS+NHL(IX,IY)-1
      DO 100 L=LS,LEND
      I=LISTH(L)
      QX=(X-XC(I))/SX(I)
      QXSQ=QX*QX
      IF  (QXSQ .GE.9.) GO TO 100
      QY=(Y-YC(I))/SY(I)
      QYSQ=QY*QY
      IF  (QYSQ .GE. 9.) GO TO 100
      QXY=TWORHO(I)*QX*QY
      FACTOR=SCALE(I)*(QXSQ+QYSQ+QXY)
      IF(FACTOR.LT.-3.) GO TO 100
      HT=PEAK(I)*EXP(FACTOR)
      IF(HT.LE.ZMAX) GO TO 100
      ZMAX = HT
  100 CONTINUE
  150 TMAC=ZMAX
      RETURN
      END
```

```
      SUBROUTINE STOCH(I,RANGE,A)
C
C     SUBROUTINE TO COMPUTE STOCHASTIC ATTRITION COEFFICIENT
C
      COMMON /GRP6/ ALPHA(6)
      COMMON /GRP3/ NBU,NRU,FL(6),FO(6),NOI(3),XIC(3,200),YIC(3,200),
     1IDIR(3,200),AVSP,ISPD
     1,IUSTAT(6),II(6),LOST(6,6),VISFRA,VISFRB,SIZETK,
     1SIZETW,NT(6),NF(6),SRF,DISMAX,
     1NLOSC(6,6),VISFR(6,6),RMINTK,RMXTK,RMINTW,RMXTW,OP,TOWFR,TNKFR,
     1PTT(3,3),RF,POA(6,6),APOA(6,6),LOA(6,6),NA(6),OFL(6),POL(6)
      IF(I.EQ.2) GO TO 5003
      A=ALPHA(I)*((1.0-RANGE/RMXTW)**2)
      GO TO 5004
 5003 A=ALPHA(I)*((1.0-RANGE/RMXTK)**2)
 5004 RETURN
       END
```

```
      SUBROUTINE ETK(I,RANGE,T)
C
C     SUBROUTINE TO COMPUTE EXPECTED TIME TO KILL A TARGET.
C
      COMMON /GRP2/ TA(2),T1(2),TH(2),TM(2),TF1(2),TF2(2),TF3(2),
     1P(2,6),PHM(2,6),PHM(2,6),PKH(2,6),TF(2)
      IF(I.EQ.2) GOTO 5
      TF(I)=TF1(I)
      GOTO 6
    5 IF(RANGE.GT.1000.0) GOTO 7
      TF(I)=TF1(I)-(TF1(I)*(1000.0-RANGE)/1000.0)
      GOTO 6
    7 IF(RANGE.GT.2000.0) GOTO 8
      TF(I)=TF2(I)-((TF2(I)-TF1(I))*(2000.0-RANGE)/1000.0)
      GOTO 6
    8 TF(I)=TF3(I)-((TF3(I)-TF2(I))*(3000.0-RANGE)/1000.0)
    6 J=(RANGE+250.0)/500.0
      IF(J.GT.6) J=6
      T=TA(I)+T1(I)-TH(I)+((TH(I)+TF(I))/PKH(I,J))+((TM(I)+TF(I))/
     1PHM(I,J))*((1.0-PHM(I,J))/PKH(I,J)+PHH(I,J)-P(I,J))
      RETURN
      END
```

```
C
      SUBROUTINE SORT(I,M)
      COMMON /GRPS/ LOT(6,6),ROT(6,6)
      DO 19 J=1,M
      IF(ROT(I,M).GE.ROT(I,J)) GOTO 19
   21 R=ROT(I,J)
      NN=LOT(I,J)
      ROT(I,J)=ROT(I,M)
      LOT(I,J)=LOT(I,M)
      ROT(I,M)=R
      LOT(I,M)=NN
   19 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE KOVER(ZO,THACT,SIZET,ZT,S,HTS,ZS,VISFRT)
C
      IF(S.EQ.0.) GO TO 2000
      IF(HTS.GE.ZS) GO TO 2050
      HEXT=ZO+(HTS-ZO)/S
      EVIST=AMAX1(HEXT,THACT)
      IF(EVIST.GE.ZT) GO TO 2050
      IF(EVIST.LE.ZT-SIZET) RETURN
      VIS=(ZT-EVIST)/SIZET
      IF(VIS.LT.VISFRT) VISFRT=VIS
      RETURN
 2000 IF(HTS.LT.ZO) RETURN
 2050 VISFRT=0.0
      RETURN
      END
```

APPENDIX C

Definition of Variables in Computer Program

ALPHA(I)	= Initial attrition-rate coefficient for stochastic attrition module.
APOA(I,J)	= The average proportion of the j^{th} attacker of unit i allocated to fire on unit i.
AVSP	= Average speed of moving attacking units.
BREAK	= Breakpoint distance between attackers and defenders.
DISMAX	= Maximum distance allowed between attacking units before the leading unit is delayed.
DIST	= The straight-line distance between two movement nodes inputed by the user.
DST	= The distance to be moved each time step by attacking units.
FL(I)	= Force level of unit i.
FO(I)	= Initial force level of unit i.
IALT	= Denotes whether the user desires alternate defensive positions or not.
IC	= Counts number of time units a defender has been moving.
IDIR(I,J)	= Direction of j^{th} interval in i^{th} route.
II(I)	= Interval index for unit i.
IMOVE	= Number of time units a defender is allowed for moving to an alternate position.
IPRDIR(I)	= Primary direction of movement for unit i.
IRTE	= Denotes whether user wants to input routes or not.
IS	= Random number seed used for stochastic attrition.
ISECWD(I)	= Width of search sector for unit i.

ISPD = Input variable to denote user's desired speed for attackers movement.

ITEM = Input variable denoting number of time steps allowed for defender's move.

ITIME = Current time, in seconds, of battle.

ITRIT = Input variable denoting whether attrition will be stochastic or deterministic.

IUSTAT(I) = Current status of unit i.

LCA(I,J) = The number of the j^{th} attacker of unit i.

LOST(I,J) = Denotes whether line-of-sight exists between unit i and j or not.

LCT(I,J) = The number of the j^{th} target of unit i.

MVTDIR(I) = Movement direction of unit i.

N(I) = Number of nodes inputed by user for route i.

NA(I) = Number of attackers of unit i.

NBU = Number of blue units.

NF(I) = Number of time units unit i is allowed to fire at the same location.

NLCSC(I,J) = Number of continous time steps that line-of-sight does not exist between unit i and unit j.

NOI(I) = Number of intervals in the i^{th} route.

NRU = Number of Red Units.

NT(I) = Number of targets of unit i.

OFL(I) = Force level of unit i during previous time step.

P(I,J) = Probability of 1^{st} round hit by unit i in range band j.

PHH(I,J) = Probability of a hit following a hit by unit i in range band j.

PHM(I,J) = Probability of a hit following a miss by unit i in range band j.

PKH(I,J) = Probability of a kill given a hit by unit i in range band j.

PM = The proportion of time a moving unit is searching for targets.

PCA(I,J) = The proportion of the j^{th} attacker of unit i allocated to fire on unit i.

PCL(I) = Percent of unit i lost during the current time step.

PTT(I,J) = Proportion of surviving firepower allocated to the i^{th} target if there are j targets available.

RANGE = Current minimum distance between attackers and defenders.

RMINTK = Minimum effective range for attacking weapon system.

RMINTW = Minimum effective range for defending weapon system.

RMXTK = Maximum effective range for attacking weapon system.

RMXTW = Maximum effective range for defending weapon system.

RCT(I,J) = The range of the j^{th} target of unit i.

SIZETK = Size of attacking vehicle.

SIZETW = Size of defending vehicle.

TA(K) = Time to acquire a target for k^{th} weapon system type (k = 1,2).

TF1(K) = Time of flight to 1000m for k^{th} weapon system type (k = 1,2).

TF2(K) = Time of flight to 2000m for k^{th} weapon system type (k = 1,2).

TF3(K) = Time of flight to 3000m for k^{th} weapon system type (k = 1,2).

TH(K)	= Time to fire a round following a hit for weapon system type k (k = 1,2).
TI(K)	= Time to fire first round after target has been acquired for weapon system type k (k = 1,2).
TM(K)	= Time to fire a round following a miss for weapon system type k (k = 1,2).
TNKFR	= Firing rate for attacking weapon system.
TOWFR	= Firing rate for defending weapon system.
TPOL(I)	= Total percentage of lost since battle began for unit i.
VISFR(I,J)	= The fraction of unit i seen by unit j.
VISFRA	= Fraction of unit A as seen by unit B.
VISFRB	= Fraction of unit B as seen by unit A.
X(I),Y(I)	= Coordinates of unit i.
XA(I),YA(I)	= Coordinates of alternate position for defender i.
XIC(I,J) YIC(I,J)	= Coordinates of the j^{th} interval endpoint of the route for unit i.
XL,YL	= Distance added to previous interval endpoint for vehicle to move DST during a time step.
XLOC(I,J) YLOC(I,J)	= Coordinates of the j^{th} node inputed by the user for the route of unit i.

BIBLIOGRAPHY

Bonder, S., "The Lanchester Attrition-Rate Coefficient," Operations Research, v. 15, p. 221-232, May 1967.

Bonder, S., "The Mean Lanchester Attrition Rate," Operations Research, v. 18, p. 179-181, June 1970.

Clark, G. M., The Combat Analysis Model, Ph.D. Thesis, Ohio State University, Columbus, Ohio, 1969.

Karr, A. F., Stochastic Attrition Models of Lanchester Type, P-1031, Institute for Defense Analysis, Arlington, Virginia, June 1974.

Smoler, J., Operational Lanchester-Type Model of Small Unit Land Combat, M.S. Thesis, Naval Postgraduate School, Monterey, California, 1979.

Taylor, J. G., Force-on-Force Attrition Modelling, p. 20-89, Military Applications Section of Operations Research Society of America, January 1980.

Taylor, J. G., "Solving Lanchester-Type Equations for 'Modern Warfare' with Variable Coefficients," Operations Research, v. 22, p. 756-770, October 1974.

Wallace, W. S., and Hagewood, E. G., Simulation of Tactical Alternative Responses (STAR), M. S. Thesis, Naval Postgraduate School, Monterey, California, December 1978.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 55 Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
4.	Professor James G. Taylor, Code 55Tw Department of Operations Research Naval Postgraduate School Monterey, California 93940	10
5.	Professor Glenn F. Lindsay, Code 55Ls Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
6.	Professor Samual H. Parry, Code 55Py Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
7.	Captain Glenn M. Mills 710 Manhasset Road Charlotte, North Carolina 28209	2
8.	Command and General Staff College Attn: Education Advisor Room 123, Bell Hall Ft. Leavenworth, Kansas 66027	1
9.	Headquarters, Department of the Army MILPERCEN Attn: DAPC-OPF-C 200 Stovall Street Alexandria, Virginia 22332	1

DATE
ILME